UNIVERSIDADE FEDERAL DE UBERLÂNDIA
FACULDADE DE ENGENHARIA QUÍMICA
PROGRAMA DE PÓS-GRADUAÇÃO EM ENGENHARIA QUÍMICA

INTELIGÊNCIA ARTIFICIAL APLICADA À DETECÇÃO E DIAGNÓSTICO DE FALHAS EM PROCESSOS QUÍMICOS

Uberlândia

2020

UNIVERSIDADE FEDERAL DE UBERLÂNDIA
FACULDADE DE ENGENHARIA QUÍMICA
PROGRAMA DE PÓS-GRADUAÇÃO EM ENGENHARIA QUÍMICA

INTELIGÊNCIA ARTIFICIAL APLICADA À DETECÇÃO E DIAGNÓSTICO DE FALHAS EM PROCESSOS QUÍMICOS

Matheus Henrique Granzotto

Tese de doutorado apresentada ao Programa de Pós-graduação em Engenharia Química da Universidade Federal de Uberlândia como parte dos requisitos necessários à obtenção do título de Doutor em Engenharia Química, área de concentração em Pesquisa e Desenvolvimento de Processos Químicos.

Uberlândia

2020

Dados Internacionais de Catalogação na Publicação (CIP)
Sistema de Bibliotecas da UFU, MG, Brasil.

G765i
2020

Granzotto, Matheus Henrique, 1984-
Inteligência artificial aplicada à detecção e diagnóstico de falhas em processos químicos [recurso eletrônico] / Matheus Henrique Granzotto. - 2020.

Orientador: Luís Cláudio Oliveira Lopes.
Tese (Doutorado) - Universidade Federal de Uberlândia, Programa de Pós-Graduação em Engenharia Química.
Modo de acesso: Internet.
Disponível em: http://doi.org/10.14393/ufu.te.2020.3005
Inclui bibliografia.
Inclui ilustrações.

1. Engenharia química. I. Lopes, Luís Cláudio Oliveira, 1964-, (Orient.). II. Universidade Federal de Uberlândia. Programa de Pós-Graduação em Engenharia Química. III. Título.

CDU: 66.0

Rejâne Maria da Silva – CRB6/1925

UNIVERSIDADE FEDERAL DE UBERLÂNDIA
Coordenação do Programa de Pós-Graduação em Engenharia Química
Av. João Naves de Ávila, 2121, Bloco 1K, Sala 206 - Bairro Santa Mônica, Uberlândia-MG, CEP 38400-902
Telefone: (34)3239-4249 - www.ppgeq.feq.ufu.br - secppgeq@feq.ufu.br

ATA DE DEFESA - PÓS-GRADUAÇÃO

Programa de Pós-Graduação em:	Engenharia Química				
Defesa de:	Tese de Doutorado, 06/2020, PPGEQ				
Data:	30 de abril de 2020	Hora de início:	14:00	Hora de encerramento:	17:00
Matrícula do Discente:	11613EQU008				
Nome do Discente:	Matheus Henrique Granzotto				
Título do Trabalho:	Inteligência Artificial Aplicada à Detecção e Diagnóstico de Falhas em Processos Químicos				
Área de concentração:	Desenvolvimento de Processos Químicos				
Linha de pesquisa:	Modelagem, Controle e Otimização de Processos Químicos				
Projeto de Pesquisa de vinculação:	Desenvolvimento de sistemas de controle de processos tolerantes a falhas				

Reuniu-se por meio de webconferência, a Banca Examinadora designada pelo Colegiado do Programa de Pós-graduação em Engenharia Química, assim composta: Professores Doutores: Maurício Bezerra de Souza Júnior - DEQ/UFRJ; Thiago Vaz da Costa - IRN/UNIFEI; Josué Silva de Morais - FEELT/UFU; Humberto Molinar Henrique - FEQUI/UFU; Rubens Gedraite - PPGEQ/UFU e Luís Cláudio Oliveira Lopes - PPGEQ/UFU, orientador do candidato.

Iniciando os trabalhos o presidente da mesa, Prof. Dr. Luís Cláudio Oliveira Lopes, apresentou a Comissão Examinadora e o candidato, agradeceu a presença do público, e concedeu ao Discente a palavra para a exposição do seu trabalho. A duração da apresentação do Discente e o tempo de arguição e resposta foram conforme as normas do Programa.

A seguir o senhor(a) presidente concedeu a palavra, pela ordem sucessivamente, aos(às) examinadores(as), que passaram a arguir o(a) candidato(a). Ultimada a arguição, que se desenvolveu dentro dos termos regimentais, a Banca, em sessão secreta, atribuiu o resultado final, considerando o(a) candidato(a):

Aprovado.

Esta defesa faz parte dos requisitos necessários à obtenção do título de Doutor.

O competente diploma será expedido após cumprimento dos demais requisitos, conforme as normas do Programa, a legislação pertinente e a regulamentação interna da UFU.

Nada mais havendo a tratar foram encerrados os trabalhos. Foi lavrada a presente ata que após lida e achada conforme foi assinada pela Banca Examinadora.

Documento assinado eletronicamente por **Luis Claudio Oliveira Lopes**, **Professor(a) do Magistério Superior**, em 30/04/2020, às 20:26, conforme horário oficial de Brasília, com fundamento no art. 6º, § 1º, do Decreto nº 8.539, de 8 de outubro de 2015.

Documento assinado eletronicamente por **Thiago Vaz da Costa**, **Usuário Externo**, em 30/04/2020, às 20:50, conforme horário oficial de Brasília, com fundamento no art. 6º, § 1º, do Decreto nº 8.539, de 8 de outubro de 2015.

Documento assinado eletronicamente por **MAURÍCIO BEZERRA DE SOUZA JÚNIOR**, **Usuário Externo**, em 30/04/2020, às 21:12, conforme horário oficial de Brasília, com fundamento no art. 6º, § 1º, do Decreto nº 8.539, de 8 de outubro de 2015.

Documento assinado eletronicamente por **Humberto Molinar Henrique**, **Professor(a) do Magistério Superior**, em 01/05/2020, às 12:16, conforme horário oficial de Brasília, com fundamento no art. 6º, § 1º, do Decreto nº 8.539, de 8 de outubro de 2015.

Documento assinado eletronicamente por **Rubens Gedraite**, **Professor(a) do Magistério Superior**, em 04/05/2020, às 07:53, conforme horário oficial de Brasília, com fundamento no art. 6º, § 1º, do Decreto nº 8.539, de 8 de outubro de 2015.

Documento assinado eletronicamente por **Josué Silva de Morais**, **Professor(a) do Magistério Superior**, em 04/05/2020, às 11:32, conforme horário oficial de Brasília, com fundamento no art. 6º, § 1º, do Decreto nº 8.539, de 8 de outubro de 2015.

A autenticidade deste documento pode ser conferida no site https://www.sei.ufu.br/sei/controlador_externo.php?acao=documento_conferir&id_orgao_acesso_externo=0, informando o código verificador **2016438** e o código CRC **1C815CC8**.

Referência: Processo nº 23117.028911/2020-41 SEI nº 2016438

Dedico esta tese a minha família pela fé e confiança demonstrada.
Aos meus amigos pelo apoio incondicional.
Ao orientador pela paciência demonstrada no decorrer do trabalho.
A Carol e Éric, que estão comigo presentes em todos os momentos.
Enfim a todos que de alguma forma tornaram este caminho mais fácil de ser percorrido.

AGRADECIMENTOS

Ao amigo e orientador Prof. Dr. Luís Cláudio Oliveira Lopes por todo o conhecimento passado, pelas excelentes supervisões, orientações, pelas diretrizes seguras e permanente incentivo.

Agradeço a Deus, por me presentear de diversas formas ao longo da vida e me permitir vivenciar essa experiência única de crescimento e aprendizado.

Agradeço aos meus pais, Alexandre e Marlene por todos os sacrifícios que fizeram em prol de minha educação e formação. Pela confiança e apoio demonstrados para que eu concluísse este trabalho.

Agradeço a minha esposa, Carolyne, pelo apoio e amor incondicionais, por ser essa companheira fiel, amiga em todas as horas, sempre dedicada, amorosa e preocupada. Muitas vezes rigorosa, mas sempre pensando no meu melhor como pessoa e como profissional. Obrigado por ter suportado a distância e ter segurado todas as "pontas" sozinha, pelas ajudas e noites em claro do meu lado e por nunca duvidar de minha capacidade até quando eu mesmo duvidei. Você é meu amor, a luz da minha vida.

Agradeço ao meu filho, Éric, que hoje com um ano já é o maior amor de minha vida, presente divino, razão de todos os meus esforços.

Agradeço a Dona Neyde, minha vó-sogra, por ser essa figura tão maternal em minha vida. Sempre carinhosa, compreensiva e zelosa. Seus cafés e sanduíches, sempre prontos para a minha chegada, não sei se sou capaz de retribuir todo o amor e apoio que me dá e confiança que deposita em mim. Minha eterna gratidão à senhora por ser tanto em minha vida.

Agradeço aos meus irmãos, em especial ao meu irmão Gabriel, por toda a parceria e as palavras de apoio, incentivo e afeto que sempre diz. Saiba que é você o meu exemplo.

Agradeço a UFVJM pelo período de aperfeiçoamento pessoal.

À Coordenação de Aperfeiçoamento de Pessoal de Nível Superior (CAPES) pela bolsa de estudos de Doutorado.

RESUMO

Métodos de detecção de falhas são utilizados para detectar operações anormais em processos químicos provocadas por perturbações, degradação de equipamentos, defeitos nos sensores e nos equipamentos. Esses métodos podem ser baseados em: análise de resíduos, redundância analítica ou redundância física. Os métodos com redundância analítica estão entre os mais utilizados industrialmente. A redundância analítica pode ser obtida de três formas: através do modelo matemático analítico, do modelo de identificação ou inteligência artificial por aprendizado de máquina. Os métodos de detecção de falhas utilizando inteligência artificial têm sido cada vez mais presentes na literatura na última década. Os métodos que utilizam redes neuronais, lógica difusa e máquinas de vetores de suporte, são técnicas de detecção de falhas com larga utilidade e confiabilidade. O procedimento de detecção é feito em três etapas: treinamento do modelo de detecção, com a utilização de dados históricos do processo, normais e anormais; validação do modelo; e detecção de padrões de falhas. Atualmente, a extração automática de características relevantes ao procedimento de detecção e classificação de padrões de operações anormais ao processo tem se mostrado uma tendência dentro do âmbito da detecção de falhas. O presente trabalho consiste no estudo e análise dos métodos de detecção de falhas baseados em aprendizagem de máquina (redes neuronais, lógica difusa e máquinas de vetores de suporte) e a síntese de novos métodos de detecção de falhas baseado em aprendizado profundo. Aprendizado profundo (*deep learning*) compreende abordagem inovadora em aprendizado de máquina no sentido de dispensar parte do pré-processamento necessário aos métodos clássicos de detecção de falhas. Isto ocorre ao gerar de forma automática propriedades invariantes nas suas camadas de representação hierárquicas. Estes métodos têm apresentado resultados promissores em diferentes aplicações tais como o reconhecimento de voz e processamento de linguagem natural. Os algoritmos de aprendizado profundo podem solucionar problemas encontrados em metodologias de detecção de falhas, principalmente, na extração de propriedades e pré-processamento de dados. Os resultados obtidos pela metodologia proposta de redes neuronais linguísticas apontam que métodos de detecção de falhas com aprendizado podem ser facilmente aplicados nos mais diferentes processos, somente com a utilização de dados históricos e/ou modelos do processo, resultando em detecções compatíveis com os outros métodos estudados.

Palavras-chave: Detecção de falhas. Controle de processos químicos. Inteligência artificial. Máquinas de vetores de suporte. Redes neuronais artificiais. Lógica difusa. Aprendizado profundo. Redes neuronais linguísticas.

ABSTRACT

Fault detection methods are used to detect abnormal operations in processes caused by disturbances, equipment degradation, defects in sensors and equipment. These methods can be based on residue analysis, analytical redundancy, or physical redundancy. The methods with analytical redundancy are among the most industrially used approaches. The analytical redundancy in processes can be obtained through analytical mathematical models of the process through the identification of models or artificial intelligence through machine learning. Failure detection methods through artificial intelligence have been increasingly studied in the literature in the last decade. The methods that use neural networks, fuzzy logic and support vector machines are fault detection techniques with high usability and reliability. The detection procedure is done in three steps: model training for detection, using normal and abnormal process data; validation of the model; and fault pattern detection. Currently, one of the paradigms in the area of fault detection is the extraction of important characteristics to the procedure of detection and patterns classification of process abnormal operations. The present work investigates these methods of fault detection based on machine learning and proposes a new method based on deep learning. Deep learning is an innovative approach to machine learning in order to dispense part of the necessary preprocessing of classic machine learning methods by automatically generating invariant properties in its hierarchical representation layers. These methods have presented promissing results in different applications such as speech recognition and natural language processing. Deep-learning algorithms can solve problems found in fault-detection methodologies, especially in the extraction of properties and preprocessing of data. The results obtained by the proposed technique linguistics neural network show that learning failure detection methods can be easily applied in different processes, only with the use of historical data and/or process models, resulting compatible detections with other studied methodologies.

Keywords: Fault detection, process control, artificial intelligence, support vector machine, neural nets, fuzzy logic, deep learning, linguistics neural networks.

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

ANN	*Artificial Neural Networks* (Redes Neuronais Artificiais).
ANN-C	Redes Neuronais Artificiais de Classificação.
ANN-R	Redes Neuronais Artificiais de Regressão.
BIB	Blocos Incompletos Balanceados.
BICV	Validação cruzada incompleta balanceada.
BMU	*Best Matching Neuron* (Neurônio Vencedor).
CSTR	*Continuous Stirred Tank Reactor* (Reator Contínuo de Tanque Agitado).
DF/F	Razão entre número de intervalos detectados como determinada falha e o número de intervalos em que realmente ocorrem a falha.
DF/N	Razão entre número de intervalos detectados como determinada falha e o número de intervalos em que realmente ocorrem a operação normal.
DL	*Deep Learning* (Aprendizado Profundo).
DN/F	Razão entre número de intervalos detectados como operação normal e o número de intervalos em que realmente ocorrem a falha.
DN/N	Razão entre número de intervalos detectados como operação normal e o número de intervalos em que realmente ocorrem a operação normal.
DNN	*Deep Neural Network* (Rede Neuronal "Profunda").
DP	*Differential Pressure*.
FCM	*Fuzzy C-means*.
FDD	*Fault Detection and Diagnosis* (Detecção e Diagnóstico de Falhas).
FF	*Feedforward* (Alimentação direta).
FSCL	*Frequency-Sensitive Competitive Learning* (Aprendizado Competitivo Sensível a Frequência).
FSVM	*Fuzzy Support Vector Machines* (Máquinas de Vetores de Suporte *Fuzzy*).
GB	Gigabyte.
GHz	Giga Hertz.
GPU	*Graphics Processing Unit* (Unidade de Processamento Gráfico).
KKT	Karush-Kuhn-Tucker, Condições de.
LAH	*Level Alarm High*.
LAL	*Level Alarm Low*.
LIC	*Level Indicator Controller*.
LIT	*Level Indicating Transmitter*.

LNN	*Linguistic Neural Networks* (Redes Neuronais Linguísticas).
LV	*Level Valve*.
MIMO	*Multiple Input Multiple Output* (Múltiplas Entradas e Múltiplas Saídas).
MISO	*Multiple Inputs Single Output* (Múltiplas Entradas Única Saída).
ML	*Machine Learning* (Aprendizado de Máquina).
NG	*Neural Gas* (Gás Neural).
OAA	*One-Against-All* (Um Contra Todos).
OAO	*One-Against-One* (Um Contra Um).
PCA	*Principal Component Analysis* (Análise de Componente Principal).
PCM	*Perceptron* de Múltiplas Camadas.
PI	Proporcional-Integral (Controlador).
PLC	*Programmable Logic Controller* (Controladores Lógicos Programáveis).
RAM	*Random Access Memory* (Memória de Acesso Randômico).
RKHS	*Reproducing Kernel Hilbert Spaces* (Reprodução de Espaços de Núcleos de Hilbert).
RNA	Redes Neuronais Artificiais.
RPCL	*Rival-Penalize Competitive Learning* (Aprendizado Competitivo de Penalidade ao Rival).
SDCD	Sistema Digital de Controle Distribuído.
SNN	*Shallow Neural Network* (Rede Neuronal "Rasa").
SOM	*Self-Organizing Map* (Mapa Auto-Organizável).
SRM	*Structural Risk Minimization* (Minimização de Risco Estrutural).
SV	*Support Vectors* (Vetores de Suporte).
SVC	*Support Vector Classification* (Vetores de Suporte de Classificação).
SVM	*Support Vector Machines* (Máquinas de Vetores de Suporte).
SVR	*Support Vector Regression* (Vetores de Suporte de Regressão).
TAD	Tempo decorrido até detecção.
TAH	*Temperature Alarm High*.
TE	*Temperature Element* (Sensor).
TIC	*Temperature Indicator Controller*.
TIT	*Temperature Indicating Trasmitter*.
TV	*Temperature Valve*.
VC	Vapnik-Chervonenkis, dimensão de.
VLSI	*Very Large Scale Integration* (Integração em Escala Muito Grande).
WTA	*Winner-Take-All* (Vencedor Leva Tudo).

WTM	*Winner-Take-Most* (Vencedor Leva a Maioria).

LISTA DE SÍMBOLOS

$\hat{f}_{n,l}$	Função gerada pela máquina de aprendizado, baseando-se nos pares (**x**,*y*) utilizados no treinamento que minimiza o erro sobre os dados l a serem classificados no espaço de hipótese S_n.
$\bar{\alpha}_i$	Multiplicador de Lagrange *i* do SVM de regressão.
$\|\cdot\|$	Norma euclidiana.
$\|\cdot\|$	Valor absoluto.
$\langle \mathbf{w}, \mathbf{x} \rangle$	Produto interno entre **w** e **x**.
$\bar{\mathbf{w}}$	Vetor de pesos do hiperplano de regressão do SVM.
S_n	Espaço de hipótese *n*.
$\bar{b}$	Coeficiente linear do hiperplano de regressão do SVM.
$\hat{f}$	Função gerada pela máquina de aprendizado, baseando-se nos pares (**x**,*y*) utilizados no treinamento que minimiza o erro sobre os dados a serem classificados.
u_i^-	Restrição inferior da variável de entrada manipulada *i*.
u_i^+	Restrição superior da variável de entrada manipulada *i*.
$\hat{y}$	Variável obtida através de modelo, variável de referência.
A	Limite arbitrário para a norma de **w**.
a	Limite inferior de normalização.
A	Reagente ciclopentadieno (reação de van der Vusse).
A_t	Área de transferência de calor (m²).
b	Coeficiente linear do hiperplano, $\|b\|/\|\mathbf{w}\|$ = distância perpendicular do hiperplano à origem.
b	Limite superior de normalização.
B	Produto desejado ciclopentanol (reação de van der Vusse).
b^{ij}	Coeficiente linear do hiperplano entre as classe *i* e *j* do SVM OAO.
b^m	Coeficiente linear do hiperplano da classe *m* do SVM OAA.
C	Parâmetro de restrição (penalidade) do SVM.
C	Subproduto ciclopentanodiol (reação de van der Vusse).
C_i	Concentração da saída do reator do composto *i* (mol/L).
C_{if}	Concentração na corrente de alimentação do reator do composto *i* (mol/L).
$\mathbf{c}_k$	Centroide do *k*-ésimo *cluster*, *fuzzy* c-*means*.
C_p	Calor específico [kJ/(kg.K)].

C_{pw}	Calor específico do refrigerante.
$\boldsymbol{D}$	Subproduto ciclopentadieno (reação de van der Vusse).
$d(\mathbf{w},b;\mathbf{x})$	Distância do ponto **x** para o hiperplano (**w**,*b*).
$dist_j^{(n)}$	Distância euclidiana entre o vetor de entrada $\mathbf{x}^n$ em relação ao vetor de pesos w^j.
E_i	Energia de ativação da reação *i* [].
e_i	Erro.
F	Vazão de saída do reator (L/h).
$f(s)$	Função de ativação da ANN.
F_f	Vazão da corrente de alimentação do reator (L/h).
F_i	Classe de falha *i*.
F_w	Vazão da corrente de alimentação de refrigerante na camisa do reator (L/h).
h	Dimensão VC (Vapnik-Chervonenkis).
H	Hessiana.
I	Índice de desempenho.
$J(.)$	Jacobiana.
$K(\mathbf{x},\mathbf{x}')$	Função núcleo (*kernel*).
k_{Ci}	Constante proporcional do controlador PI da variável controlada *i*.
k_i	Constante de velocidade específica da reação *i*.
k_{i0}	Fator pré-exponencial da reação *i*.
l	Número de dados amostrais.
$L(f(\mathbf{x}),y)$	Resultado da escolha da função de custo relacionando a previsão $f(\mathbf{x})$ quando a saída desejada é *y*.
m_w	Massa específica do refrigerante (kg/L).
n	número de conjuntos total.
N_c	Raio de vizinhança (ANN SOM).
n_e	Número de conjuntos de estimativa.
n_t	Número de conjuntos de treinamento.
n_{tt}	Número de conjuntos de teste.
n_v	Número de conjuntos de validação.
P	Vetor de probabilidades, saída da ANN de classificação.
$P(\mathbf{x},y)$	Distribuição de Probabilidade, relação entre os dados e seus rótulos.
Q_w	Taxa de calor removido na camisa do reator (kJ/h).
R	Constante universal dos gases perfeitos.
r	Resíduo, diferença entre variável real e variável de referência.

$R(f)$	Risco da função f.
$R_{emp}(f)$	Risco empírico da função f.
R_{he}	Raio da hiper-esfera circundando todas as amostras.
r_i	Taxa da reação i.
s	Indícios analíticos.
S_i	Espaço de hipótese de dimensão i, $i = 1, 2, \ldots, \infty$.
s_j	Saída do combinador linear devido aos sinais de entrada.
T	Temperatura da corrente de saída do reator (K).
t	Tempo.
T_0	Temperatura do reator inicial (K).
T_f	Temperatura da corrente de alimentação ao reator (K).
T_w	Temperatura da corrente de alimentação de refrigerante a camisa do reator (K).
U	Coeficiente global de transferência de calor [kJ/(h.m².K)].
u	Variável de entrada do processo, variável manipulada.
u_i	Variável de entrada manipulada i.
V	Volume do reator (L).
W	Matriz de partição ($w_{i,j}$).
$\mathbf{w}$	Vetor de pesos ajustáveis normal ao hiperplano.
$\mathbf{w}_{i+1}$	Peso sináptico atualizado para a época i+1.
$\mathbf{w}^{ij}$	Vetor de pesos do hiperplano entre as classe i e j do SVM OAO.
w_{jn}	Pesos sinápticos do neurônio j.
$\mathbf{w}^m$	Vetor de pesos do hiperplano da classe m do SVM OAA.
$\mathbf{X}$	Dado a ser normalizado.
x	Variável de estado.
$\mathbf{x}_i$	Vetor contendo dados amostrais i, para $i = 1, \ldots, l$.
$\mathbf{X}_{máx}$	Valor máximo dos dados a serem normalizados.
$\mathbf{X}_{min}$	Valor mínimo dos dados a serem normalizados.
$\mathbf{X}_n$	Dado normalizado.
x_n	Sinal de entrada n.
x_r	Dados de treinamento da classe r.
x_s	Dados de treinamento da classe s.
y	Variável de saída do processo, variável medida.
$Y(t)$	Variável dependente do tempo.
y_i	Rótulo relativo ao dado amostral $\mathbf{x}_i$; Variável de saída controlada i.

y_j	Sinal de saída do neurônio *j*.
$Y_{máx}$	Valor máximo da variável *Y*.
Y_{min}	Valor mínimo da variável *Y*.
α_i	Multiplicador de Lagrange de *i*.
β_i	Multiplicador de Lagrange.
δ	Probabilidade ($0 < \delta < 1$).
ΔH_{ri}	Calor de reação da reação *i* (kJ/mol de *j*) (*j* = reagente da reação *i*).
θ	Parâmetro do modelo.
λ	Parâmetro de regularização ($\lambda = 1/2C$).
μ_i	Função de pertinência *fuzzy* de *i*.
ξ_i	Medida dos erros de classificação dos dados *i*.
ξ_i^-	Variável folga inferior.
ξ_i^+	Variável folga superior.
ρ	Massa específica da solução do reator (kg/L).
$\rho(\mathbf{w},b)$	Margem do hiperplano ($\mathbf{w}$,*b*).
σ	Desvio padrão.
τ_{Ii}	Constante integral do controlador PI da variável controlada *i*.
$\mathcal{D}$	Hiperplano.
$\phi(\mathbf{w})$	Função a ser minimizada pelo SVM.
$\mu(x)$	Função de pertinência *Fuzzy*.

SUMÁRIO

CAPÍTULO 1
INTRODUÇÃO

A utilização de métodos de detecção de falhas em controle de processos é atualmente um campo em desenvolvimento, uma vez que a demanda crescente por qualidade dos produtos, eficiência, redução de custos e integração dos sistemas de controle no setor industrial em inúmeros processos químicos, cujos custos elevados envolvidos e a necessidade sempre presente de segurança, justificam a importância do monitoramento e dos sistemas de detecção e diagnóstico de falhas envolvendo tais processos (ISERMANN, 2006). Para monitorar um processo, ou seja, checar se variáveis determinadas ultrapassaram seus respectivos limites, antigamente eram utilizadas metodologias clássicas, que atualmente podem ser substituídas por sistemas mais elaborados baseados em várias técnicas modernas provenientes da inteligência artificial, do reconhecimento de padrões e da modelagem de sistemas.

Os métodos de detecção de falhas baseados em reconhecimento de padrões são exemplos de métodos que, se mais desenvolvidos, podem auxiliar enormemente o controle de plantas químicas no reconhecimento de padrões diferentes das operações normais. As máquinas de vetores de suporte, as redes neuronais e a lógica difusa têm a capacidade de tornar controles automáticos de processos químicos mais eficientes encontrando padrões de falhas conhecidas (classificação) ou reconhecendo desvios do comportamento das variáveis de controle (identificação).

1.1. Motivação

A partir da década de 1960 (ISERMANN, 2006) a automação industrial aplicada à operação e projeto de processos, teve um grande crescimento devido à utilização de computadores. Com o intuito de reduzir custos, houve a diminuição da presença de operadores (humanos) na execução dos processos químicos, bem como a não utilização destes operadores em tarefas repetitivas e monótonas. Posteriormente, em 1975, ocorreu um significativo aumento no nível de automação envolvido nos processos industriais, uma vez que o crescimento na disponibilidade dos microcomputadores contribuiu para o aumento de sistemas controlados automaticamente e, consequentemente, de sistemas para detecção de falhas. Esse desenvolvimento foi seguido pelo avanço na área de atuadores, sensores e interfaces máquina-usuário. O aprofundamento da teoria existente sobre processos e funções de controle na detecção de falhas também contribuiu na busca por melhores métodos.

A tecnologia envolvida nos processos industriais atuais resulta em um meio cada vez mais preciso, eficiente, confiável e seguro, comparando estes processos com os mesmos encontrados no passado. O monitoramento do processo produtivo proporcionado pelo uso de tecnologias vindas da eletrônica digital e da informática, permite o monitoramento dos produtos desde a sua fase inicial de produção até o seu contato com o mercado, o que proporciona um controle maior de todos os eventos do processo de fabricação de produtos industrializáveis, sejam estes produtos químicos ou não (SOBHANI-TEHRANI; KHORASANI, 2009).

São expressivas as motivações para aumentar o grau de controle em processos industriais, mesmo com toda a tecnologia já disponível. O rigor das leis ambientais, o desperdício de reagentes e produtos, possíveis prejuízos financeiros devido à danos em equipamentos e a concorrência cada vez maior em diversos setores da indústria justificam a busca por melhores métodos de controle afim de evitar perdas financeiras (ISERMANN, 2006).

Além do aumento do controle e do monitoramento de processos, a tecnologia presente cria condições para disponibilizar informações detalhadas sobre os processos químicos. Estas informações podem ser utilizadas em sistemas auxiliares que possuem a intenção de detectar possíveis falhas em controladores desses processos (GERTLER, 1998).

A contribuição deste trabalho se efetiva no campo de detecção de falhas, considerando que, através deste, é possível beneficiar-se com a descoberta em tempo hábil para que o sistema de controle possa reagir a possíveis problemas dentro dos processos industriais. A utilização deste método prevê a diminuição de perdas consideráveis, tais como: paralisação de processos, danos a equipamentos de alto custo e de fundamental importância, acidentes envolvendo vidas humanas, dentre outras. Essas perdas podem ser manifestadas pelas perdas financeiras de montantes significativos a depender do processo em questão.

Os sistemas de detecção e identificação de falhas (*Fault Detection Identification*, FDI) são geralmente implementados tendo como objetivo a detecção e classificação das falhas conhecidas. A detecção da falta no processo é a constatação de que há uma modificação inesperada no comportamento do sistema que degrada o desempenho ou desconstrói a operação normal do processo. Uma falha é usualmente o resultado da progressão de uma ou mais faltas que podem ocasionar situações de perigo, ou seja, a falta constitui a gênese da falha. Consequentemente, a detecção da falha é a constatação de que há algo errado que poderá levar o processo a algum desvio significativo do seu funcionamento normal. A verificação da existência de anormalidade no comportamento do sistema se dá através da comparação efetuada entre os dados provenientes da planta física e os valores estimados por modelos matemáticos do processo (VENKATASUBRAMANIAN *et al*., 2003a, 2003c).

Utiliza-se como ferramenta da classificação de falhas um algoritmo denominado classificador. Este analisa os sinais provenientes da detecção de falhas, também conhecidos como resíduos, resultando na classificação, ou não, de uma falha.

O presente trabalho aborda algumas das metodologias utilizadas para realizar a detecção e classificação de falhas através de máquinas de vetores de suporte (VAPNIK, 1998), redes neuronais artificiais (GALUSHKIN, 2007) e lógica difusa (*Fuzzy*) (ZADEH, 1965). O sistema de detecção de falhas deve ser capaz de se adaptar e detectar diferentes falhas as quais um processo está sujeito. Para isso, o sistema se baseia nos sinais de entrada e saída do processo real, além do modelo matemático ou de dados históricos que servirão como referência para a identificação das possíveis anormalidades.

1.2. Objetivos

O objetivo geral deste trabalho é propor e analisar o desempenho de metodologias de detecção de falhas com inteligência artificial e aprendizado profundo.

Como objetivos específicos, é proposto:

- Analisar e estudar máquinas de vetores de suporte, redes neuronais artificiais e lógica *fuzzy*;
- Analisar e avaliar metodologia de detecção e diagnóstico de falhas utilizando máquinas de vetores de suporte;
- Analisar e avaliar metodologia de detecção e diagnóstico de falhas utilizando redes neuronais artificiais;
- Analisar e avaliar metodologia de detecção e diagnóstico de falhas utilizando lógica *fuzzy*;
- Propor e analisar metodologias de detecção e diagnóstico de falhas utilizando aprendizado profundo;
- Propor e analisar metodologia híbrida de detecção e diagnóstico de falhas *fuzzy*-ANN;

1.3. Contribuições

São diversas as contribuições trazidas por este trabalho. Dentre elas, podem ser citados como mais relevantes aquelas que fazem referência à metodologia de detecção e diagnóstico de falhas proposta:

- Detecção das faltas e falhas em processos industriais indicadas pelos modelos e sinais de entrada e saída do sistema;

- Capacidade de adaptação da metodologia reunindo informações relevantes ao longo da operação do processo, formando uma base de dados para detecções posteriores;
- Capacidade de detecção das falhas conhecidas que são adicionadas anteriormente a base de dados;
- Capacidade de aplicação em ambientes simulados resultando na efetivação da utilização da metodologia mesmo diante da presença de ruído;
- Utilização de conceitos de aprendizado profundo em algoritmos de detecção e diagnóstico de falhas.

1.4. Organização

Este trabalho foi organizado de maneira a facilitar a compreensão do leitor. De modo global, foram analisados e apresentados métodos de detecção de falhas baseados em dados históricos de processos químicos com a utilização de inteligência artificial, para isso foram apresentados conceitos pertinentes a temática, um breve histórico sobre detecção de falhas, suas teorias e respectivas aplicações dentro de estudos de casos seguindo as metodologias de máquinas de vetores de suporte, redes neuronais artificiais e lógica difusa. Finaliza-se com a apresentação e comprovação da aplicação de proposta de nova metodologia de detecção de falhas tendo como cerne a utilização de algoritmos de aprendizado profundo.

Desta forma, tem-se inicialmente no Capítulo 2, um breve histórico de detecção de falhas em controle de processos dinâmicos, seus principais conceitos e as teorias concernentes, bem como, as diferentes categorias de metodologias que já foram anteriormente implementadas dentro do campo de detecção de falhas.

No Capítulo 3 realizou-se descrição detalhada da metodologia utilizada no trabalho para a obtenção e tratamento dos dados históricos que serão utilizados nos sistemas de detecção e diagnóstico de falhas, foram apresentados também os índices de desempenho para comparação das diferentes técnicas de detecção de falhas utilizadas.

A metodologia das máquinas de vetores de suporte (*Support Vector Machines* - SVM) em classificação (*Support Vector Classification*) e detecção (*Support Vector Regression*), suas teorias, respectivas aplicações no campo de detecção e diagnóstico de falhas e os resultados obtidos nos estudos de casos estão contidas no Capítulo 4.

A organização dos Capítulos 5 e 6, denominados respectivamente Detecção de Falhas com Redes Neuronais Artificiais e Detecção de Falhas Através de Lógica Difusa é semelhante à do Capítulo 4 ao tratarem sobre a teoria e aplicação de redes neuronais artificiais (*Artificial Neural Networks* - ANN) e lógica difusa (*Fuzzy*) em detecção de falhas, juntamente com os resultados dos estudos de casos para cada metodologia apresentada.

No desenvolvimento do Capítulo 7 são explicados os conceitos de aprendizado profundo e também demonstrados os algoritmos propostos de detecção de falhas com aprendizado profundo, suas aplicações e respectivos resultados obtidos dentro dos estudos de casos utilizados neste trabalho. Há também dentro deste capítulo, o estudo acerca da comparação entre os algoritmos propostos e as demais metodologias anteriormente apresentadas.

Por fim, no Capítulo 8, encontram-se as conclusões do presente trabalho, bem como propostas para novos trabalhos.

CAPÍTULO 2
DETECÇÃO DE FALHAS EM CONTROLE DE PROCESSOS

2.1. Introdução

Este capítulo aborda uma revisão de metodologias de detecção de falhas de controle: são apresentadas diversas características das metodologias de detecção de falhas, como as máquinas de vetores de suporte, redes neuronais, entre outras.

O objetivo final de um algoritmo de detecção de falhas de controle em processos químicos é detectar um padrão anormal de informações fornecidas pelo sistema de controle, que pode provocar a curto, médio ou longo prazo, uma degradação do controle do processo, fazendo com que a qualidade da produção caia, o processo se torne instável ou a segurança do processo seja comprometida. Este objetivo é perseguido, de forma geral, verificando a modificação das variáveis do processo medido ao longo do tempo, em comparação com modelos obtidos a partir das informações obtidas através da operação normal do processo, ou mesmo de informações de falhas obtidas previamente. O algoritmo é capaz de calcular a distância em tempo real das variáveis de controle para os modelos treinados pela metodologia de detecção de padrões utilizada, como as máquinas de vetores de suporte.

2.2. Controle de Processos Industriais

O incremento na complexidade dos processos industriais e confiabilidade requerida em seu funcionamento tornam necessários grandes avanços na área de controle. Estes avanços permitem resolver uma grande gama de problemas na indústria, mas não são suficientes para garantir uma boa operação. Além de controlar, é necessário também assegurar o correto funcionamento do processo (). As funções de supervisão servem para indicar estados indesejados ou não permitidos no processo, e proceder a ações para manter a operação e evitar danos ou acidentes. As seguintes funções devem ser diferenciadas (ISERMANN, 1997):

a) Monitoramento: variáveis medidas são checadas em relação a sua tolerância, e alarmes são gerados para o operador.
b) Proteção automática: no caso de estados perigosos do processo, a função de monitoramento inicia automaticamente uma ação apropriada.
c) Supervisão com diagnóstico de falhas: baseando-se nas variáveis medidas, cálculos são feitos, falhas são diagnosticadas e procedimentos para recuperação de falhas são adotados.

Os métodos clássicos a) e b) são adequados para supervisão geral do processo. Para ajustar as tolerâncias, compromissos devem ser estabelecidos entre os tamanhos dos desvios e a necessidade dos alarmes, uma vez que as variáveis medidas têm flutuação normal. A verificação de limites é uma boa solução quando o processo opera próximo ao estado de regime permanente. Caso o ponto de operação varie rapidamente, a situação exige procedimentos mais elaborados. No caso de sistema em malha fechada, a atuação do controle pode mascarar situações anormais do processo, não permitindo sua detecção a partir das variáveis de saída.

Portanto, métodos avançados para supervisão e diagnóstico de falhas são necessários para satisfazer:

- Detecção imediata de pequenas falhas;
- Diagnóstico de falhas em atuadores, sensores e componentes do processo;
- Detecção de falhas em processos em malha fechada.
- Supervisão de processos em regime transitório.

O objetivo da detecção e diagnóstico imediatos é ter tempo hábil para ações de reconfiguração, manutenção e proteção, e pode ser conseguida coletando informações que permitam estabelecer relações entre variáveis do processo. Para o diagnóstico de falhas, o conhecimento das relações causa-efeito é necessário.

Existem quatro procedimentos associados com a supervisão e diagnóstico de falhas em processos (BLANKE *et al*., 2003): detecção de falhas, identificação de falhas, diagnóstico de falhas e recuperação do processo.

Detectar falhas é determinar se uma falha ocorreu. Detecções adiantadas disponibilizam informações valiosas em problemas emergenciais, permitindo ações apropriadas para evitar problemas sérios no processo.

Identificar falhas é selecionar as variáveis observadas mais relevantes para diagnosticar a falha. O propósito desse procedimento é chamar a atenção do operador ou engenheiro da planta para o subsistema a qual a falha é mais pertinente levando a uma ação para eliminação da falha mais apurada (ISERMANN, 2006).

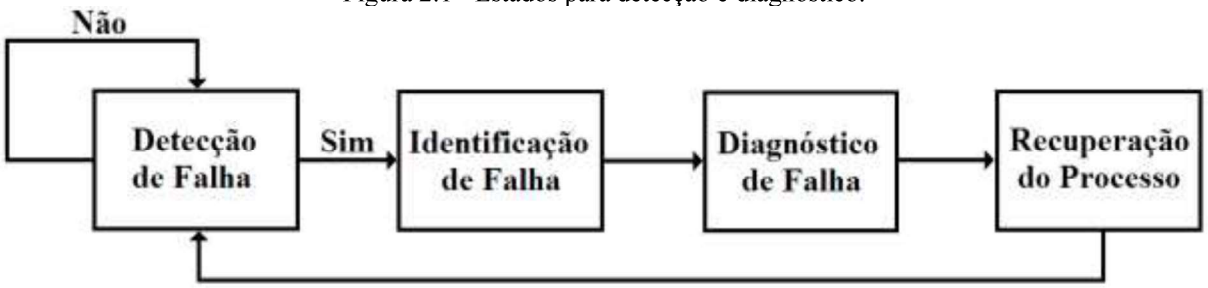
Diagnosticar a falha é determinar qual falha ocorreu, dizendo a causa da anormalidade observada. Ou ainda, diagnóstico de falha é determinar o tipo, o local, a magnitude e o momento da falha. Esse procedimento é essencial para contra-atacar ou eliminar a falha.

A recuperação do processo ou intervenção envolve remover o efeito da causa, voltando o processo para o seu estado normal.

A Figura 2.1 representa a malha fechada para monitoramento de processos. Apesar dos quatro procedimentos serem sequenciais, não necessariamente todos eles devem ser automatizados. Por exemplo, a automação dos dois primeiros procedimentos permitirá ao

operador e engenheiro do processo fazer o diagnóstico e intervir no processo de forma a recuperar o mesmo antes que algum problema ocorra.

Figura 2.1 - Estados para detecção e diagnóstico.



Fonte: adaptado de Chiang *et al*. (2001).

Para detecção de falhas, usualmente, utilizam-se limites nas variáveis e quando um desses limites é ultrapassado uma falha é detectada. Fazendo medidas de variáveis que caracterizam o comportamento do processo, e comparando o valor medido de uma variável com as demais é possível determinar a variável mais afetada pela falha. Porém esse tipo de detecção limita a possibilidade de diagnóstico de falhas tanto de forma automática como de forma manual, pois cabe ao operador ou engenheiro da planta toda a interpretação e isolação das falhas encontradas.

Neste trabalho, a abordagem de identificação de padrões será utilizada para realizar os diagnósticos. Esta abordagem implica na existência de um modelo matemático para estimar as saídas do processo, as quais são comparadas com as saídas medidas gerando resíduos, além de informações prévias de falhas já conhecidas. A existência de resíduos indica a existência de falhas. A comparação com padrões de falhas já conhecidas torna o processo de identificação da falha automático. O método para geração de resíduos deve ser tolerante em relação aos ruídos, às incertezas nos parâmetros e às dinâmicas não modeladas. Os métodos usuais para geração de resíduos são a estimativa de parâmetros, observadores ou estimadores de estados e equações de igualdade (BLANKE *et al*., 2003).

Em processos industriais, geralmente, não existem sistemas dedicados à detecção de falhas. Os programas de supervisão de processo são utilizados para detectar alguns tipos fixos de falhas a partir de limites fixados sobre as variáveis de interesse.

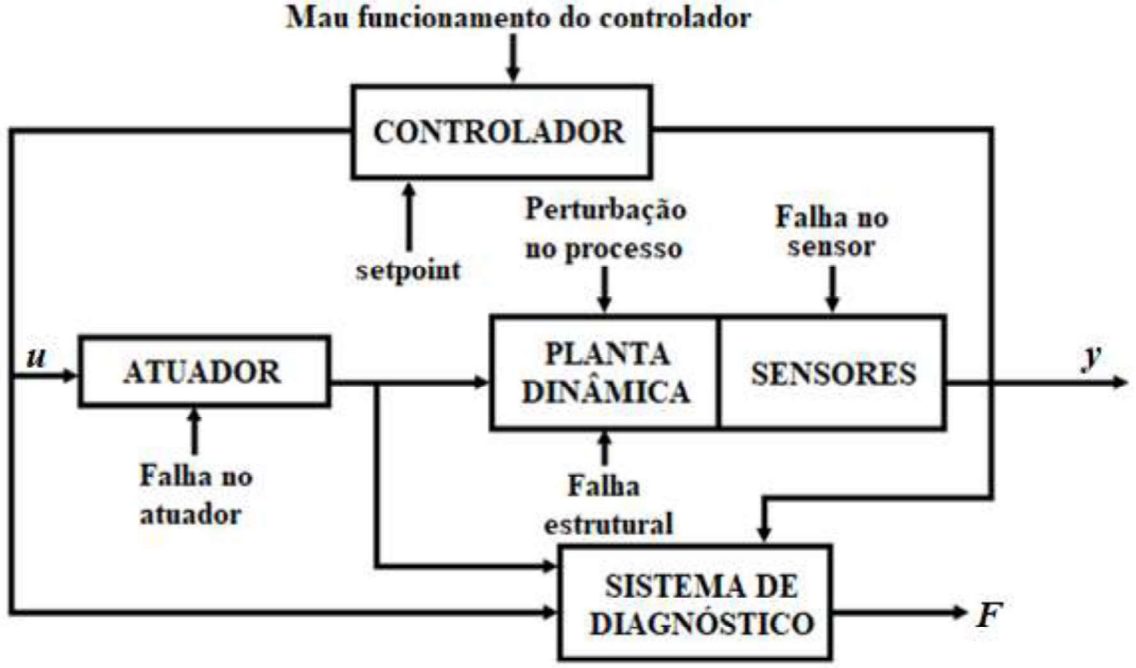
Para a detecção de falhas em processos industriais é possível utilizar a estrutura existente. Geralmente, os processos industriais são compostos de equipamentos como PLCs (*Programmable Logic Controller* - Controlador Lógico Programável) ou SDCDs (Sistema Digital de Controle Distribuído) para executar os algoritmos de controle e sistemas de supervisão para interface com o processo. Os algoritmos de controle podem ser desenvolvidos tanto nos equipamentos (PLCs e SDCDs) como nos sistemas de supervisão do processo

industrial. Existem também alguns *softwares*, tais como: Controlst, MES, PORTRAIT®, NC Monitor2, CITECTSCADA que estão disponíveis no mercado, desenvolvidos por empresas como a GE Inteligent Plataforms®, SAP®, SYSCON PlantStar®, Mitsubishi Electric®, Schneider Electric®, dentre outras diversas empresas do ramo que desenvolvem *softwares* para o monitoramento de processos químicos baseados em dados históricos REFERÊNCIAS.

2.3. Tipos de Falhas

Os principais tipos de falhas que podem ocorrer em um processo industrial são: falhas em sensores, falhas em atuadores e falhas estruturais. Na Figura 2.2 observa-se uma estrutura típica para diagnóstico de falha. A variável *u* representa as entradas do sistema de controle, *y* representa as saídas do sistema e *F* representa a classe a que determinada falha pertence.

Figura 2.2 - Estrutura típica de diagnóstico de falha.



a) Falhas em sensores

Sensores são os elementos utilizados para medição das variáveis do processo. Uma falha nesse elemento leva a uma medição errada da variável medida, acarretando possíveis ações de controle erradas.

Em processos químicos existem sensores que fazem medições vitais para controle do processo. Podem ser citados sensores de temperatura, nível, vazão e pressão. Falhas comuns que podem ocorrer em elementos, sensores ou instrumentos são a perda do sinal desse equipamento e/ou o aparecimento gradativo ou imediato de possíveis desvios na aferição do sensor. Ambas as falhas provocam ações de controle equivocadas. A perda do sinal apresenta

uma variação brusca que pode ser observada pelo operador, mas em um processo grande muitas vezes não será vislumbrada com tanta facilidade. Já o aparecimento de um desvio no instrumento, principalmente quando gradativo, dificulta a percepção por conta do operador.

b) Falhas em atuadores

Atuadores são dispositivos que convertem sinais elétricos em uma ação física. Falhas nesses dispositivos levam o processo a não responder corretamente à ação de controle, produzindo em geral grande variabilidade na qualidade dos produtos produzidos. Os atuadores mais comuns são as válvulas.

Geralmente os processos de controle de nível e temperatura são compostos por diversas válvulas. A obstrução causada pela deposição de material na região da válvula é a causa mais comum de falha nos atuadores, bem como a alteração de características da válvula, acarretando problemas de não linearidade tais como histerese e zona morta, tendo como consequência o desgaste do atuador. Outros exemplos de atuadores são motores, bombas e aquecedores. Da mesma forma que as válvulas, esses atuadores são comuns em muitos processos químicos. É comum que com o tempo ocorra um desgaste desses equipamentos o que altera os seus comportamentos. Esses desgastes podem ocasionar modificação na resposta dos atuadores e consequentemente na resposta do processo como um todo. Essas modificações interferem no processo reduzindo e/ou afetando sua qualidade.

c) Falhas estruturais

A estrutura é toda a parte física da planta, desconsiderando atuadores e sensores do processo. É responsável pela sustentação de todo o processo. Falhas estruturais são mais difíceis de serem detectadas, tais como, as falhas que envolvem obstruções, vazamentos e desgastes. Pode-se citar alguns exemplos de falhas estruturais, são elas: vazamentos ou obstruções de tubulações, dutos ou tanques.

2.4. Metodologias de Detecção de Falhas

Existem diversas técnicas que já foram utilizadas com o objetivo de detectar falhas em processos químicos. Na literatura, autores que trabalham neste tema apresentaram métodos particulares de como catalogar essas técnicas (WILLSKY; 1976; GERTLER, 1998; FRANK, 1990; LOU *et al*., 2003; VENKATASUBRAMANIAN *et al.*, 2003b, ANGELI; CHATZINIKOLAOU, 2004; ISERMANN, 2006; KATIPAMULA; BRAMBLEY, 2005a, 2005b; CASTILLO; EDGAR, 2008; MESKIN; KHORASANI, 2011; MILJKOVIĆ, 2011; MING; ZHAO, 2017). Nesta tese, escolheu-se seguir a abordagem encontrada em Isermann

(2006), por ser mais intuitiva quanto à classificação das técnicas. Em geral, os sistemas de detecção de falhas podem ser baseados tanto em redundância analítica quanto em redundância de equipamentos (*hardware*). Neste trabalho, apenas considera-se o caso de redundância analítica, ou seja, quando existem informações a respeito do modelo do sistema. São apresentadas a seguir as principais abordagens baseadas na redundância analítica.

2.4.1. Detecção de falhas com verificação de limites

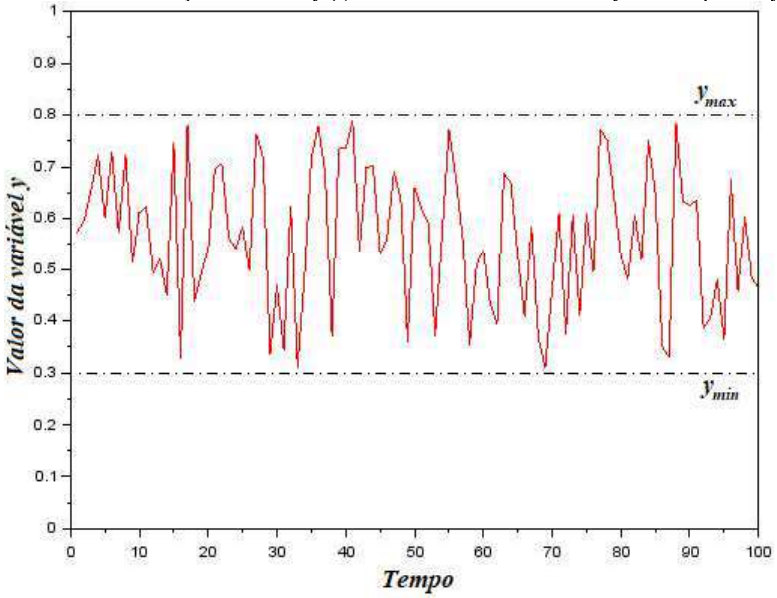
O método mais simples e utilizado frequentemente para detecção de falhas é a averiguação do limite de uma variável $Y(t)$ medida de forma direta. As variáveis de um processo são monitoradas e comparadas aos valores limites, também chamados de *thresholds*.

Geralmente, dois valores são escolhidos para serem os limites (*thresholds*) máximo Y_{max} e mínimo Y_{min}. O valor normal caracteriza-se quando (Equação (2.1)

$$Y_{min} < Y(t) < Y_{max} \tag{2.1}$$

O processo se encontra em uma situação normal se a variável monitorada se encontra dentro da faixa de tolerância. Se a variável monitorada excede um dos limites, significa que o processo se encontra em modo de falha. Esse método é aplicado à grande maioria de todos os sistemas de automação de processos e as variáveis podem ser: temperatura do processo, pressão do reator, vazão dos reagentes, entre outras. Os limites são selecionados a partir de experiências e indicam alguma margem de segurança. A Figura 2.3 mostra o monitoramento de uma variável hipotética $y(t)$ e seus respectivos limites inferior y_{min} e superior y_{max}.

Figura 2.3 - Verificação do sinal $y(t)$ entre os limites inferior y_{min} e superior y_{max}.

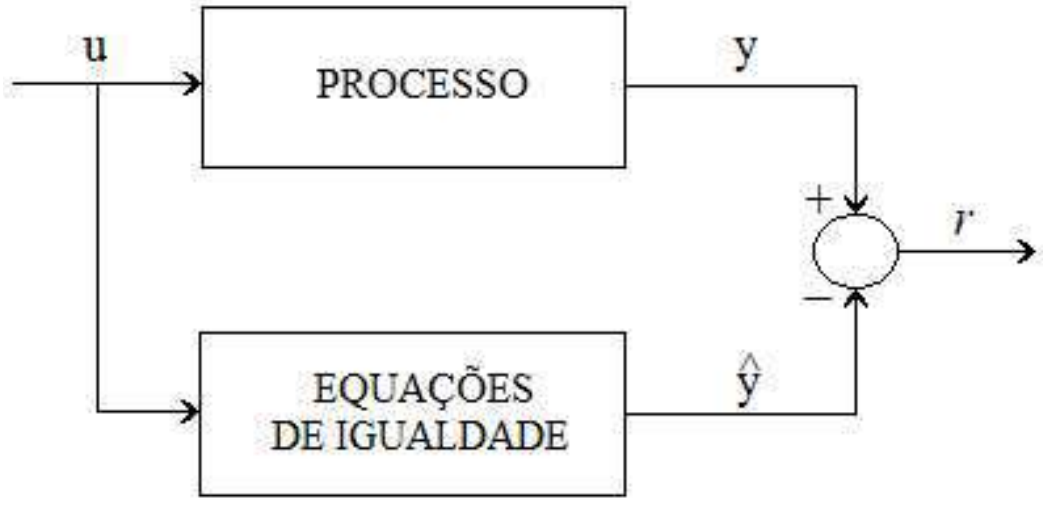


Apesar de eficiente, esse tipo de verificação é bastante simples. Entretanto, quando o sinal ultrapassa um dos limites (tanto o inferior quanto o superior) isto não necessariamente corresponde a presença de uma falha no processo. Nestes casos, os alarmes apresentam ou identificam uma falsa falha. Outro aspecto importante é que os valores dos limites podem ser funções do ponto de operação do processo, o que leva a política da programação *on-line* do limite. Isto aumentaria bastante a complexidade do procedimento.

2.4.2. Detecção de falhas com equações de igualdade

Um método simples de detecção de falhas em processos é comparar o seu comportamento real com o seu comportamento nominal, ou seja, sem estar em modo de falha. Nesse caso, a comparação dos sinais de saída do processo real com os sinais de saída das equações do mesmo processo fornece os chamados resíduos, que são sinais que serão analisados para a decisão sobre falhas ou não. A abordagem de detecção de falhas com equações de igualdade tem a capacidade de indicar anormalidades no processo através das discrepâncias entre o sinal de saída do processo e das equações. Porém, é necessário o conhecimento prévio das equações e dos parâmetros que regem o processo. Devido à dificuldade de se obter equações precisas em casos reais e das mesmas não levarem em consideração os ruídos encontrados na natureza dos processos, esse método não é muito utilizado, porém o seu princípio de funcionamento por comparação com um processo nominal livre de falhas é amplamente popular em outras técnicas. A Figura 2.4 ilustra em esquema de blocos essa abordagem, onde a variável *r* representa os resíduos.

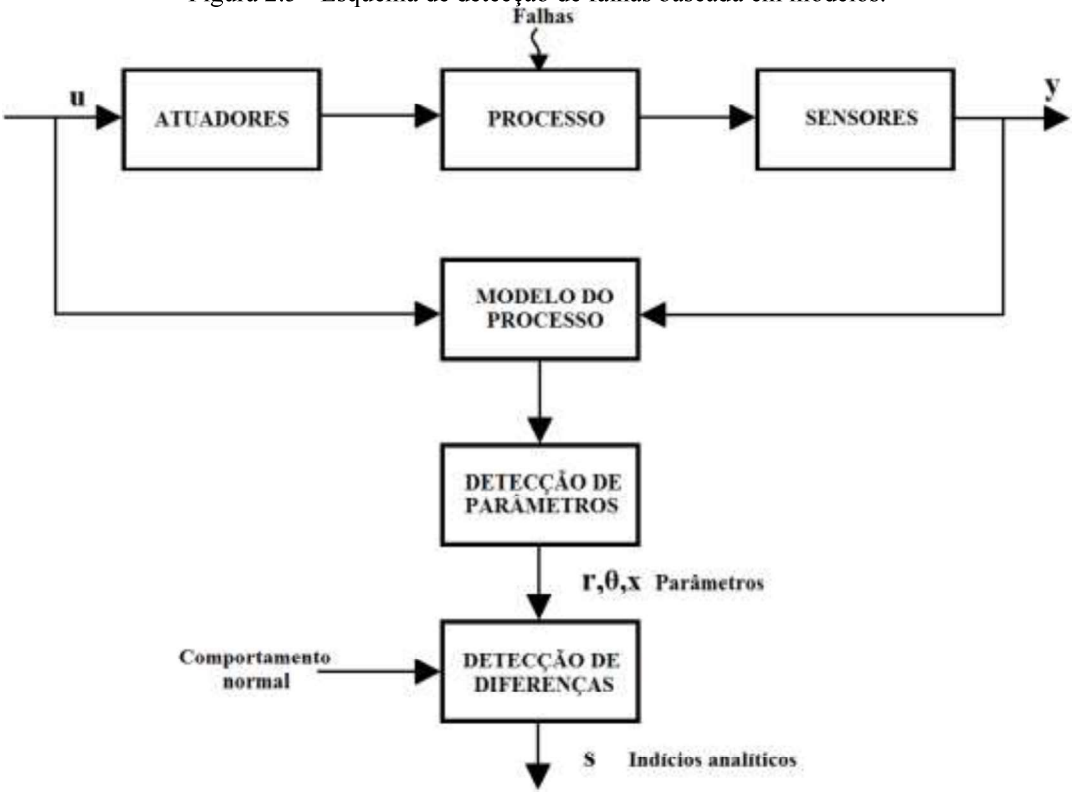
Figura 2.4 - Método de detecção de falhas por equações de igualdade - Esquema de blocos.



2.4.3. Modelos de processos e modelagem das falhas

Metodologias de detecção de falhas baseadas em modelos usam a correspondência entre variáveis advindas do modelo e do processo real para inferir informação de possíveis mudanças ocorridas por falhas. Essas correspondências são, na maior parte das vezes, relações analíticas na forma de equações dos modelos de processos. As relações entre os sinais de entrada *U* e os sinais de saída *Y* são representados pelo modelo matemático do processo (DING, 2013). A Figura 2.5 mostra o esquema básico para o sistema de detecção de falhas baseado em modelo.

Figura 2.5 - Esquema de detecção de falhas baseada em modelos.



Fonte: adaptado de ISERMANN, 2006.

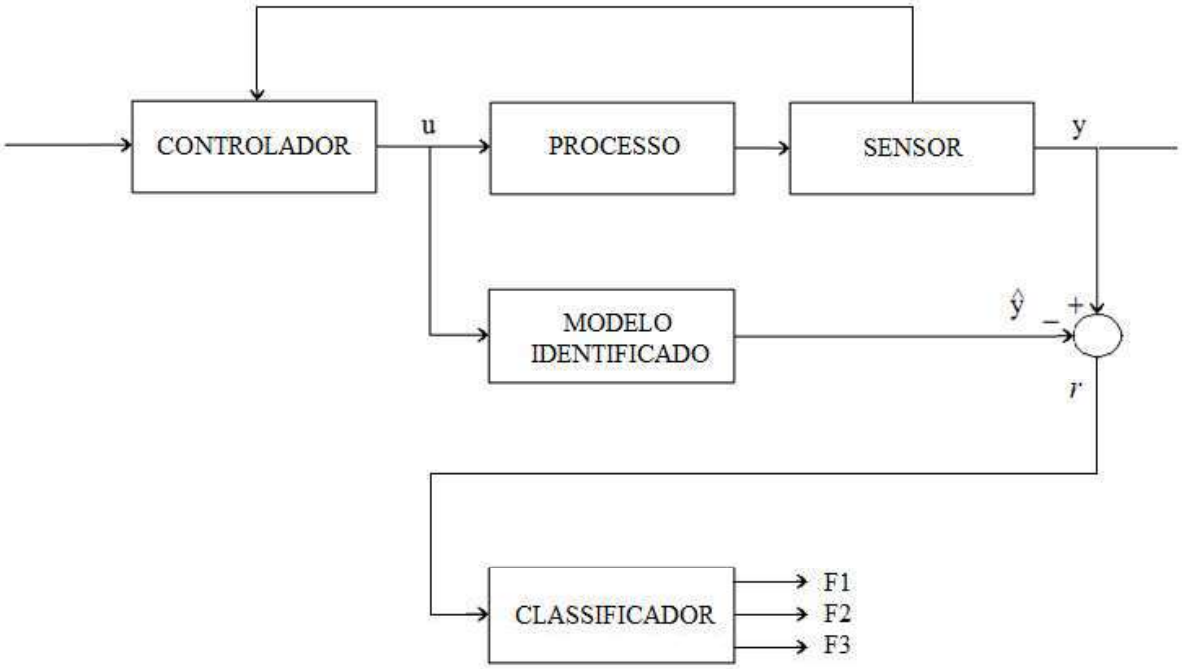
As metodologias de detecção de falhas baseadas em modelos são abalizadas pelas seguintes características: os parâmetros θ, as variáveis de estado *x* e os resíduos *r*. Comparando esses valores observados com os valores nominais, aplicando as metodologias de detecção de diferenças através da comparação das características do processo normal com as características obtidas através do modelo matemático ou identificado do processo, são gerados os indícios, ou sintomas, analíticos *s*. As modificações resultantes ou discrepâncias são consideradas os sintomas analíticos.

Esses indícios são a base para a classificação de falhas. Eles servem de entrada no processo de classificação e são diretamente responsáveis por indicar as falhas ocorridas.

2.4.4. Detecção de falhas com métodos de identificação de processos

O modelo matemático de um processo representa a relação entre as entradas ($u(t)$) e as saídas ($y(t)$) do mesmo. Os modelos são fundamentais para as metodologias de detecção de falhas através de identificação de processos, já que estes servirão como referência do processo quando o mesmo está em situação normal de funcionamento. Em diversos casos estes modelos são parcialmente conhecidos ou não são conhecidos, enquanto que em outros apenas alguns parâmetros são desconhecidos. Para a metodologia de detecção de falhas através de identificação de processos, é de extrema importância a qualidade dos modelos, que devem ser precisos a ponto de expressar detalhadamente os desvios provocados por prováveis falhas no processo. Por consequência, metodologias de identificação de processos devem ser aplicadas frequentemente antes de se aplicar qualquer metodologia de detecção de falhas. A área de identificação de sistemas possui uma extensa literatura, onde pode-se encontrar maiores informações (BILLINGS, 1980; LJUNG, 1987; VAN OVERSCHEE, 1994; FERNANDES *et al.*, 2006; AGUIRRE, 2007; CATANACH; BECK, 2017). A Figura 2.6 mostra o esquema geralmente utilizado nos processos de detecção de falhas com métodos de identificação de processos.

Figura 2.6 - Esquema de método de detecção e diagnóstico de falhas através de identificação de processos.



A Figura 2.6 mostra que o módulo designado “classificador” recebe resíduos, representados por r, gerados a partir das saídas do processo real e do modelo. Estes resíduos são a base para a classificação da falha. Enfim, a saída do classificador apresenta qual falha

ocorreu. As falhas estão representadas por *F1*, *F2* e *F3*. Neste exemplo, a etapa de classificação envolve as fases de isolamento e identificação das falhas.

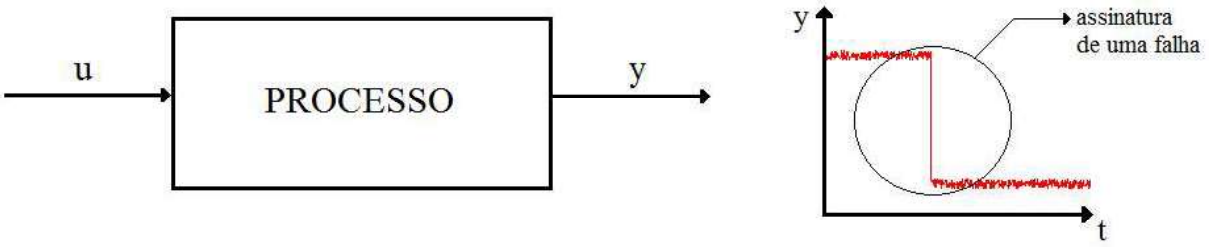
Na maioria dos casos, os modelos utilizados em metodologias de detecção e diagnóstico de falhas baseadas na identificação de processos são do tipo caixa preta, isto é, seus parâmetros não têm nenhum significado físico, nenhuma relação direta com o modelo do fenômeno do processo em questão. A técnica de detecção de falhas por identificação de processos foi utilizada por Gertler (1998), entre outros (RUSSEL *et* al., 2000; ARMENGOL *et al.*, 2003; DING *et al.*, 2005; SIMANI, 2005; FERNANDES *et al.*, 2007; SILVA *et al.*, 2007; SHEN *et al.*, 2015; OLIVIER; CRAIG, 2017).

2.4.5. Detecção de falhas com análise da assinatura de sinais

Para processos de grande proporção, como plantas químicas, o desenvolvimento de metodologias de detecção de falhas baseadas em modelo requer um grande esforço. Métodos baseados em análise de dados oferecem uma alternativa, especialmente aqueles baseados em análise estatística multivariável, como a Análise de Componentes Principais (*Principal Component Analysis* - PCA) e métodos baseados na identificação de padrões, como as Máquinas de Vetores de Suporte (*Support Vector Machines* - SVM).

Esses métodos podem ser utilizados quando as medições dos processos são altamente correlacionadas e apenas alguns eventos (falhas), produzem dados não regulares ou descorrelacionados. A Figura 2.7 ilustra o esquema da detecção de falhas por análise de sinais.

Figura 2.7 - Representação da detecção de falhas por análise de sinais.



Fonte: adaptado de SILVA, 2008.

2.4.6. Métodos de detecção de falhas baseados em inteligência artificial

Na literatura é encontrada uma extensa gama de trabalhos envolvendo diversas técnicas diferentes aplicadas à tarefa de detectar falhas em sistemas dinâmicos. Porém, além das técnicas já apresentadas anteriormente, atualmente existe uma crescente utilização de métodos baseados em inteligência artificial, empregados nas diversas abordagens de detecção de falhas (identificação de sistemas, assinatura de falhas, etc.).

As redes neuronais artificiais (*Artificial Neural Networks* - ANN) tem se tornado uma ferramenta muito utilizada na detecção de falhas em sistemas de controle. Existem diversos trabalhos na literatura que envolvem a utilização desta ferramenta (SRINIVASAN; BATUR, 1994; NAUGHTON *et al.*, 1996; TINOS; TERRA, 1998; LINARIÉ; KOROMAN, 2003; ZHANG; GE, 2015; OLIVEIRA *et al.*, 2017). Uma classificação das técnicas utilizando redes neuronais em sistemas de detecção de falhas pode ser encontrada em Madani (1999).

A lógica *fuzzy* (ou difusa) aparece nos sistemas de detecção de falha em alguns trabalhos (RICH; KNIGHT, 1994; MONSEF *et al.*, 1997; KOSCIELNY *et al.*, 1999; FRIES; GRAHAM, 2003, BALLESTEROS-MONCADA *et al.*, 2015; LI *et al.*, 2016; DHIMISH *et al.*, 2018).

A detecção de falhas, faltas ou anormalidades é uma etapa de extrema importância, pois é nela que se encontra a parte do sistema responsável por indicar se o processo está em funcionamento normal ou em modo de falha.

2.5. Estado da Arte

A detecção de falhas em processos químicos tem despertado o interesse de muitos pesquisadores de modo que já existe uma quantidade considerável de publicações sobre este tema. Entre os métodos de detecção, as máquinas de vetores de suporte, as redes neuronais e a lógica *fuzzy* foram utilizadas em diversos sistemas diferentes, além de métodos considerados clássicos, como a análise de componentes principais (*Principal Component Analysis*, PCA). Sejam em processos lineares ou não-lineares, as diferentes metodologias foram aplicadas a uma grande gama de processos.

Em processos batelada, Zhang e colaboradores (2020) contribuíram com um trabalho utilizando metodologia de detecção de falhas utilizando análise de componentes de *kernel* entrópico e máquinas de vetores de suporte multiclasses.

Sivaram e colaboradores (2020) realizaram um estudo da utilização de redes neuronais profundas para problemas de classificação.

Wang e Zhang (2019) utilizaram o PCA e as redes neuronais artificiais para um processo CSTR simulado.

Yang e colaboradores (2019) utilizaram o SVM com árvore binária para realizar a detecção de falhas em um processo químico.

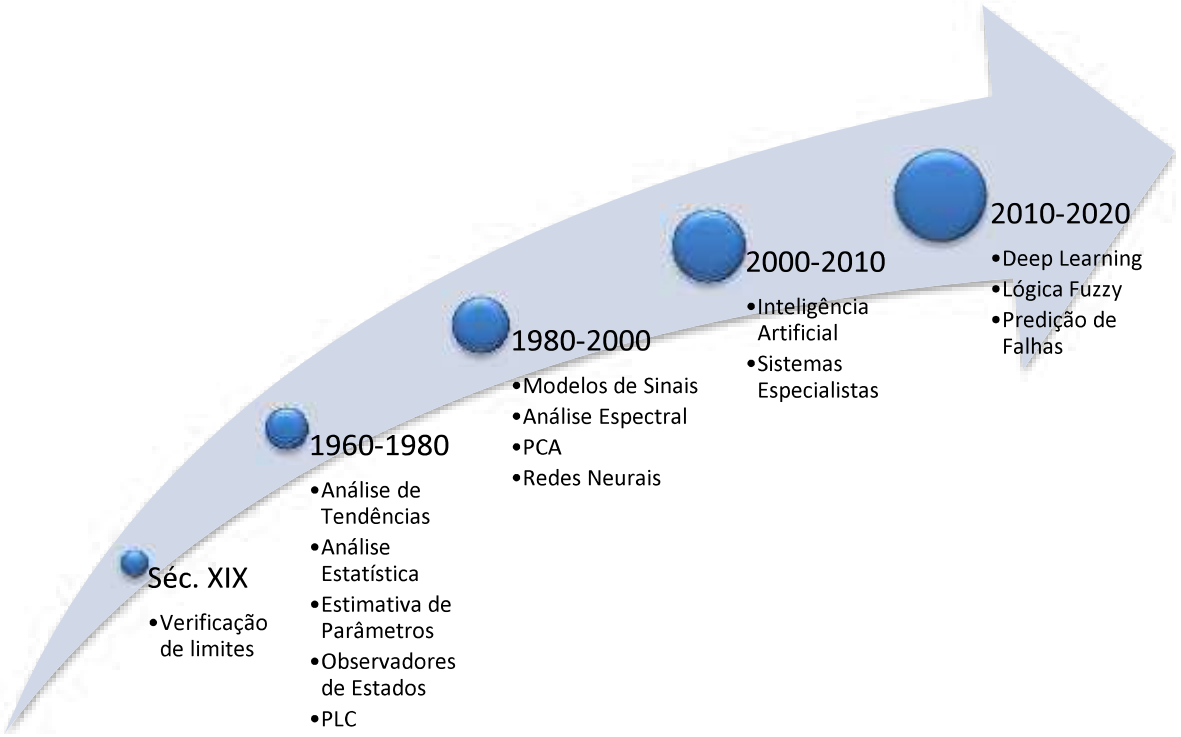
Li e Chen (2019) colaboraram com um estudo sobre algoritmos genéticos na detecção de falhas em processos químicos utilizando o PCA.

Wu e Zhao (2018) estudaram a utilização de problemas de diagnóstico de falhas em processos químicos baseados em modelos com redes convolucionais profundas.

2.6.Histórico

A Figura 2.8 apresenta uma linha do tempo simplificada das metodologias de detecção e diagnóstico de falhas a partir do desenvolvimento e utilização dos métodos de detecção e diagnóstico de falhas em processos industriais.

Figura 2.8 - Linha do Tempo - Metodologias de Detecção e Diagnóstico de Falhas.



A verificação de limites é provavelmente tão antiga quanto a instrumentação de máquinas datando do final do Século XIX. Em torno de 1960, controladores analógicos com amplificadores baseados em transistores (amplificadores operacionais) e controladores sequenciais com dispositivos *hardwired* se tornaram disponíveis e continuavam a utilizar a verificação de limites. Métodos baseados em modelos de sinais como análise espectral foram desenvolvidos com filtros análogos de passagem de banda e osciloscópios.

A implementação de computadores em processos operando *on-line* em 1960 então abriu caminho para métodos de supervisão melhorados, como a análise de tendências. Em 1968, os primeiros controladores lógicos programáveis (PLC) foram introduzidos para substituir os controladores *hardwired* por *relays* eletromecânicos, tornando a proteção de sistemas mais fácil. O advento do microcomputador em 1971 e a sua crescente aplicação em sistemas de automação de processos descentralizados desde 1975 foi o começo da supervisão de plantas e algoritmos de detecção de falhas baseada em *software*. As primeiras publicações de metodologias de detecção de falhas baseadas em modelos de processos apareceram em conexão

com sistemas aeroespaciais (BEARD, 1971; JONES, 1973; WILLSKY, 1976; CLARK, 1990) e com plantas químicas (HIMMELBLAU, 1978). Muitos destes primeiros conceitos podem ser classificados como abordagens com equações de igualdade, checando a consistência de medidas de instrumentos, de massa ou de balanços materiais. Resíduos de balanços de massa, por exemplo, são aplicados para detectar vazamentos em tubulações (SIEBERT; ISERMANN, 1977). A metodologia de detecção de falhas baseada em equações de igualdade foi aprofundada pelos estudos realizados por Gertler e Singer (1985).

As metodologias baseadas em observadores de estado foram desenvolvidas para gerar resíduos de saída, aplicando observadores de estado de Luenberger (BEARD, 1971; JONES, 1973) ou filtros de Kalman (MEHRA; PESCHON, 1971). A redundância analítica entre várias medições foi utilizada para detectar falhas em sensores, aplicando-se um grupo de observadores (CLARK, 1990). De maneira a compensar as entradas não medidas, observadores de entradas ou saídas desconhecidas foram desenvolvidos (WATANABE; HIMMELBLAU, 1982; FRANK; WÜNNENBERG, 1987). Observadores com autovalores e autovetores foram aplicados por Patton (1988).

Outra maneira de realizar a detecção de falhas é o uso da estimativa de parâmetros. As primeiras publicações surgiram em conexão com turbinas a jato (BAKIOTIS *et al*., 1979), em processos em geral como bombas de circulação e motores de corrente contínua como exemplos (ISERMANN, 1982, 1984), e em motores elétricos (FILBERT; METZGER, 1982, FILBERT, 1985)

Desde essas primeiras publicações, muitas contribuições foram realizadas na área de detecção e diagnóstico de falhas. Todo o desenvolvimento pode ser seguido por diversos artigos de revisão (ISERMANN, 1984, 1993, 1994; FRANK, 1987, 1990; GERTLER, 1988; PATTON, 1994), além de diversos livros sobre detecção de falhas (PAU, 1975; HIMMELBLAU, 1978; BRUNET *et al*., 1990; ISERMANN, 1994; BLANKE *et al*., 1997; GERTLER, 1998; CHEN; PATTON, 1999)

2.7.Indústria 4.0

A indústria 4.0, fazendo referência a quarta revolução industrial, engloba algumas tecnologias para automação e troca de dados, utilizando conceitos de Sistemas ciber-físicos, Internet das Coisas e Computação em Nuvem. Essa revolução terá um impacto mais profundo e exponencial, pois se caracteriza por um conjunto de tecnologias que permitem a fusão do mundo físico, digital e biológico. Os desafios e expectativas envolvidos nesta quarta revolução industrial são diversos, envolvendo a utilização de Inteligência Artificial, Manufatura Aditiva,

Internet das Coisas, Biologia Sintética e Sistemas Ciber-físicos, são grandes, já que o Brasil possuí um alto potencial para o desenvolvimento da quarta revolução industrial.

Os impactos da Indústria 4.0 sobre a produtividade, a redução de custos, o controle sobre o processo produtivo, a customização da produção, dentre outros, apontam para uma transformação profunda das plantas fabris. A economia envolve ganhos de eficiência, redução nos custos de manutenção de máquinas e consumo de energia.

A utilização de sistemas de detecção e diagnóstico de falhas vem de encontro aos objetivos da indústria 4.0, já que contribuem para manter em operação sistemas fabris, recuperando estes de falhas e desvios do *setpoint* desejado.

Recentemente, a segurança cibernética das redes de computadores das indústrias vem tornando-se importante na proteção de infraestruturas críticas, como plantas de geração de energia, aquedutos ou redes de transporte. Uma falha de segurança em qualquer parte desses sistemas, incluindo computadores do escritório, podem permitir que um ataque cibernético ocorra e ganhe acesso ao núcleo dos subsistemas de controle, permitindo ao atacante interferir com o comportamento correto da indústria causando danos críticos em diferentes níveis (FOVINO *et al*., 2009).

2.8.Comentários

As metodologias de detecção e diagnóstico de falhas desenvolvidas ao longo das últimas décadas possuem diversas características importantes, entre estas pode-se enumerar a capacidade de identificar falhas em um sistema defeituoso através da coleta e análise de dados sobre o estado do sistema, utilizando medições, testes e outras fontes de informação, como por exemplo, sintomas de mau funcionamento. Porém, para que os sistemas de detecção e diagnóstico de falhas sejam eficientes, é importante que os dados sejam organizados, pré-tratados, normalizados e/ou rotulados, dependendo das técnicas estudadas. Muitas técnicas baseadas em modelos ou dados históricos necessitam que o treinamento seja feito a partir de dados rotulados, como por exemplo as redes neuronais ou as máquinas de vetores de suporte de classificação. Para rotular essas amostras é necessário um conhecimento prévio dos comportamentos das variáveis e/ou processos, fazendo com que os dados normais e/ou anormais possuam os rótulos corretos e sejam adequadamente treinados, obtendo-se modelos de detecção precisos e eficientes.

Um dos esforços na área de detecção de falhas tem sido a busca pela obtenção automática das características capazes de identificar e classificar as diferentes falhas de processo. Algumas metodologias vêm sendo desenvolvidas para resolver esta questão, porém com resultados ainda intangíveis. A utilização de aprendizado profundo pode ser uma solução,

tornando o processo de obtenção das características dos dados uma parte do processo de detecção e diagnóstico de falhas. Nos próximos capítulos serão apresentadas metodologias de detecção e diagnóstico de falhas que podem ser modificadas a ponto de integrar a essas técnicas os algoritmos de aprendizado profundo, para resolver este problema.

CAPÍTULO 3
METODOLOGIA

3.1.Introdução

No presente trabalho serão realizadas simulações de processos químicos de interesse da Engenharia Química, com o objetivo de avaliar sistemas de detecção e diagnóstico de falhas (*Fault Detection and* Diagnosis, FDD), que serão apresentados ao longo deste trabalho. Ao final de cada capítulo serão apresentados os resultados obtidos para as respectivas metodologias FDD do estudo de caso apresentado na Seção 3.5. Nas próximas seções são apresentadas as técnicas utilizadas para a comparação dos diversos métodos FDD e como foram realizados os tratamentos dos dados. No capítulo de resultados e discussões serão comparados os resultados para as diversas metodologias FDD.

3.2.*Software* e *Hardware*

Este trabalho tem caráter teórico-computacional, em que se utilizou *software* matemático para a obtenção dos dados através de simulação e implementação de algoritmos de detecção e diagnóstico de falhas. O *software* livre utilizado para as simulações, obtenção e tratamento dos dados, bem como a implementação das metodologias de detecção e diagnóstico de falhas foi o Scilab® 5.5.2, na plataforma Windows.

A plataforma utilizada para as simulações foi um PC Intel® Core™2 Duo E8400 3.00 GHz, 4 GB de memória RAM, Sistema Operacional Windows 7 64 Bits, GPU GeForce 7200 GS, com driver versão 309.08.

Alguns pacotes existentes no Scilab foram adaptados para utilização nesse trabalho, dentre eles destacam-se: ANN Toolbox 0.4.2.5 (construção de redes neurais artificiais); libsvm Toolbox 1.4.5 (solução de problemas de otimização); Fuzzy Toolbox 0.4.6 (construção de funções de pertinência *fuzzy*); e Scilab Neural Network Module 2.0 (construção de redes neuronais não supervisionadas).

3.3.Índices de Desempenho

Propõem-se índices para avaliação das metodologias de detecção e diagnóstico de falhas, através da contabilização da quantidade de instantes em que a planta realmente se manteve em operação normal ou em falha e a quantidade de instantes em que o sistema de detecção constatou o estado normal ou o estado de falha. Esses índices foram calculados para

cada uma das operações anormais apresentadas para os estudos de caso, para as metodologias de detecção e diagnóstico de falhas estudados e propostos (Equação (3.1)).

$$\frac{DF}{F} = \frac{\text{instantes detectados op. falha } i}{\text{instantes corretos op. falha } i}, \quad \frac{DN}{N} = \frac{\text{instantes detectados op. normal}}{\text{instantes corretos op. normal}}$$
$$\frac{DN}{F} = \frac{\text{instantes detectados op. normal}}{\text{instantes corretos op. falha } i}, \quad \frac{DF}{N} = \frac{\text{instantes detectados op. falha } i}{\text{instantes corretos op. normal}} \tag{3.1}$$

em que: DF = quantidade de instantes detectados como falha i;

DN = quantidade de instantes detectados como operação normal;

F = quantidade de instantes em que ocorreu de fato a falha i;

N = quantidade de instantes em que ocorreu de fato a operação normal.

Em alguns casos, quando falhas diferentes das falhas simuladas foram detectadas pelo sistema de detecção, foi necessário a criação de um outro índice, apresentado na Equação (3.2, que quantifica a quantidade de erros de detecção de falha.

$$\frac{DF_i}{F_j} = \frac{\text{instantes detectados falha } i}{\text{instantes corretos falha } j} \tag{3.2}$$

em que: DF_i = quantidade de instantes detectados como Falha i;

F_j = quantidade de instantes em que ocorreu de fato a Falha j, sendo $i \neq j$.

3.4.Tratamento dos Dados

De maneira a preparar os dados para o treinamento de cada técnica de detecção e diagnóstico de falhas, existe a necessidade de normalizar ou escalonar estes, para que a variação das diferentes características não tenham maior influência sobre as outras. Os dados de entrada e saída são normalizados utilizando a Equação (3.3).

$$\mathbf{X}_n = a + (b - a) \cdot [(\mathbf{X} - \mathbf{X}_{min})/(\mathbf{X}_{max} - \mathbf{X}_{min})] \tag{3.3}$$

em que $\mathbf{X}_n$ são os valores normalizados dos dados e $\mathbf{X}_{min}$ e $\mathbf{X}_{max}$ são os valores mínimo e máximo entre todos os valores dos dados, a é o limite inferior de normalização e b é o limite superior de normalização.

Para a realização da normalização dos dados foram estabelecidos limites inferiores e superiores de 0 e 1, respectivamente, guardando informações dos menores e maiores valores

para cada uma das variáveis.

Os dados normalizados então são organizados como dados de entrada e saída, para o treinamento do método de detecção de falhas. São utilizados em torno de 70% dos dados para treinamento e em torno de 30% dos dados para validação do método. Esses dados são escolhidos de maneira que todos os dados dependentes do tempo sejam representados em ambos os grupos, treinamento e validação. A seleção de dados para treinamento e validação é realizada da seguinte forma: a cada três dados históricos que são extraídos do sistema, dois deles são utilizados no treinamento e o terceiro é utilizado para a validação. Isso se dá de maneira consecutiva até que todos os dados históricos disponíveis sejam utilizados.

Em seguida, será apresentado o estudo de caso referente ao processo de produção de ciclopentanol a partir da reação com cinética de van der Vusse.

3.5.Estudo de Caso 1 - Processo de Produção de Ciclopentanol

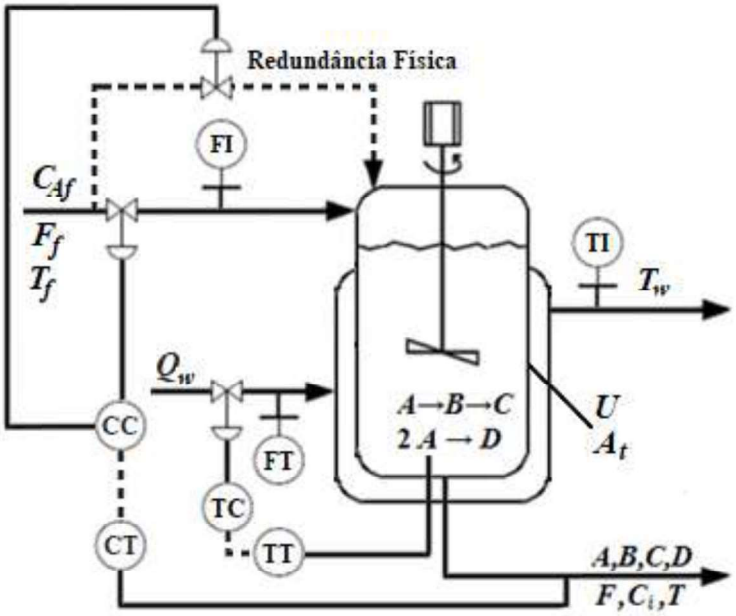
O primeiro estudo de caso utilizado para aplicação das técnicas de detecção de falhas consiste em um reator CSTR com a cinética de van der Vusse. O esquema de reação de van der Vusse tem sido frequentemente utilizada como um problema de referência para os algoritmos de controle de processos não-lineares (KLATT; ENGELL, 1998; CHEN *et al*., 1999). Este modelo apresenta comportamento não-linear dependendo dos pontos de operação considerados e é útil do ponto de vista da análise de desempenho da estratégia proposta. A complexidade relativamente baixa do sistema permite seu uso para comparação, sem efeitos de distorção, provocados pela redução do modelo. O modelo é também interessante do ponto de vista da aplicação industrial, pois representa uma classe de CSTR com resfriamento. A Figura 3.1 mostra esquematicamente o reator de mistura perfeita com a respectiva malha de controle PI proposta.

O reator é alimentado com uma solução de ciclopentadieno (A) com concentração C_{Af} para a formação de ciclopentanol (B) através da adição eletrofílica de água catalisada por ácido em solução diluída . As reações químicas no reator seguem o mecanismo de van der Vusse, em que a reação principal $A \rightarrow B$ é acompanhada por uma reação em série $B \rightarrow C$ e uma reação paralela $2A \rightarrow D$ (Equação (3.4)):

$$\begin{matrix} A \xrightarrow{k_1} B \xrightarrow{k_2} C \\ 2A \xrightarrow{k_3} D \end{matrix} \tag{3.4}$$

O ciclopentanol (*B*) é o produto desejado e o ciclopentanodiol (*C*) e o di-ciclopentadieno (*D*) são subprodutos não desejados. A reação de van der Vusse é realizada em um reator contínuo de tanque agitado (CSTR) não-isotérmico.

Figura 3.1 - CSTR com resfriamento e reação de van der Vusse.



Fonte: adaptado de KLATT; ENGELL, 1998.

O modelo do reator, considerando volume e propriedades físicas constantes, compreende o balanço de massa para cada componente e os balanços de energia do reator e da camisa (Equação (3.5)):

$$\begin{aligned}
\frac{dC_A(t)}{dt} &= \frac{F}{V}\left[C_{Af} - C_A(t)\right] + \frac{1}{V}(-r_1 - 2r_3) \\
\frac{dC_B(t)}{dt} &= -\frac{F}{V}[C_B(t)] + \frac{1}{V}(r_1 - r_2) \\
\frac{dT(t)}{dt} &= \frac{F}{V}\left[T_f - T(t)\right] + \frac{UA_t}{\rho C_p V}[T_w(t) - T(t)] - \frac{(r_1\Delta H_{r1} + r_2\Delta H_{r2} + r_3\Delta H_{r3})}{\rho C_p} \\
\frac{dT_w(t)}{dt} &= \frac{1}{m_w C_{pw}}\{Q_w + UA_t[T(t) - T_w(t)]\}
\end{aligned} \tag{3.5}$$

Nestas equações: C_{Af} é a concentração do componente A na corrente de alimentação;

F é a vazão de alimentação do reator;

V é o volume do reator;

C_p é o calor específico;

ΔH_{ri} é o calor de reação;

U é o coeficiente global de transferência de calor;

A_t é a área de transferência de calor;

r_i, i = 1, 2 ou 3, são as taxas de reação dadas por (Equação (3.6)):

$$\begin{aligned} r_1 &= k_1 C_A(t) \\ r_2 &= k_2 C_B(t) \\ r_3 &= k_3 C_A^2(t) \end{aligned} \tag{3.6}$$

As cinéticas de reação são modeladas usando expressões de Arrhenius (Equação (3.7)):

$$k_i(T) = k_{i0} exp\left(\frac{-E_i}{RT}\right), i = 1, 2\ ou\ 3 \tag{3.7}$$

As concentrações de A e B e as temperaturas do reator T e do refrigerante T_w constituem o estado $x = [C_A\ C_B\ T\ T_w]^T$ do sistema dinâmico não-linear (Equação (3.5)). Os parâmetros do modelo estão relacionados na Tabela 3.1. No esquema apresentado na Figura 3.1, foi considerado controle perfeito do nível (volume constante) e controle perfeito da quantidade de energia removida através da camisa do reator.

Tabela 3.1 - Parâmetros do modelo do reator de produção de ciclopentanol (KLATT; ENGELL, 1998).

Parâmetros	
$k_{10} = 1{,}287 \times 10^{12}$ 1/h	$\Delta H_{r1} = 4{,}2$ kJ/mol de A
$k_{20} = 1{,}287 \times 10^{12}$ 1/h	$\Delta H_{r2} = -11{,}0$ kJ/mol de B
$k_{30} = 9{,}043 \times 10^{9}$ 1/h	$\Delta H_{r3} = -41{,}85$ kJ/mol de A
$E_1/R = 9758{,}3$ K	$C_{Af} = 5{,}1$ mol/L
$E_2/R = 9758{,}3$ K	$T_0 = 130{,}0$ °C
$E_3/R = 8560$ K	$C_{pw} = 2{,}00$ kJ/(kg.K)
$\rho = 0{,}9342$ kg/L	$U = 4032$ kJ/(h.m².K)
$C_p = 3{,}01$ kJ/(kg.K)	$m_w = 5{,}0$ kg
$V = 10{,}0$ L	$A_t = 0{,}215$ m²

A vazão de entrada F_f e a taxa de energia removida Q_w servem como variáveis manipuladas (Equação (3.8)).

$$u_1 = x_3 = F_f, \qquad u_2 = x_4 = Q_w \tag{3.8}$$

Supõe-se que a temperatura do reator, a concentração do ciclopentadieno e ciclopentanol na saída do reator, a temperatura da camisa de resfriamento, a vazão de alimentação de reagente e de líquido refrigerante são obtidas através de instrumentos de medição. Para efeito de simulação do processo real, acrescenta-se aos dados simulados um

ruído gaussiano aleatório com média 0 e variância 10^{-5} para as concentrações e ruído gaussiano aleatório com média 0 e variância 10^{-3} para as demais medidas.

De acordo com Ogunnaike e Ray (1994), para uma taxa constante de calor removido e uma variação de 50 a 1500 L/h de alimentação de reagente do reator, esse processo existe em seis regiões com diferentes graus de não linearidade.

Portanto, para nosso trabalho, foram estabelecidas as seguintes restrições na forma de limites inferior e superior para a vazão de alimentação de reagente do reator e para a taxa de calor removido pelo refrigerante (Equação (3.9)):

$$\begin{aligned} u_1^- = 50\ \text{L/h} \leq F_f = u_1 \leq 350\ \text{L/h} = u_1^+ \\ u_2^- = -8500\,\text{kJ/h} \leq Q_w = u_2 \leq 0\,\text{kJ/h} = u_2^+ \end{aligned} \tag{3.9}$$

A concentração do produto C_B e a temperatura do reator T são uma escolha natural para a saída controlada (Equação (3.10)):

$$x_1 = C_B,\ \ x_2 = T \tag{3.10}$$

O controle da concentração do produto, C_B, é especificado de tal forma que qualquer valor na faixa de 0,7 mol/L à 0,95 mol/L pode ser alcançado sem desvio de estado estacionário para todos os valores de concentrações de entrada não medidas, que é nominalmente C_{Af} = 5,1 *mol/L*, mas pode variar em uma faixa de 1 mol/L a 5,7 mol/L.

A entrada dos sistemas de detecção de falhas é $\mathbf{x} = [C_B, T, F_f, Q_w]^T$.

3.5.1. Características do Processo

Para uma taxa de calor removida constante e uma variação da alimentação de reagente entre 50 L/h e 1500 L/h, Ogunnaike e Ray (1994) encontraram para este processo seis regiões com diferentes graus de não linearidade. A vazão de alimentação, no presente trabalho, foi restringida entre 50 e 350 L/h para não ultrapassar os limites físicos proporcionais ao volume do reator simulado. Para o controle do processo foram projetados dois controladores PI (Proporcional-Integral), o primeiro destinado a controlar a concentração de saída de ciclopentanol (C_B) e o segundo para a temperatura do reator (T).

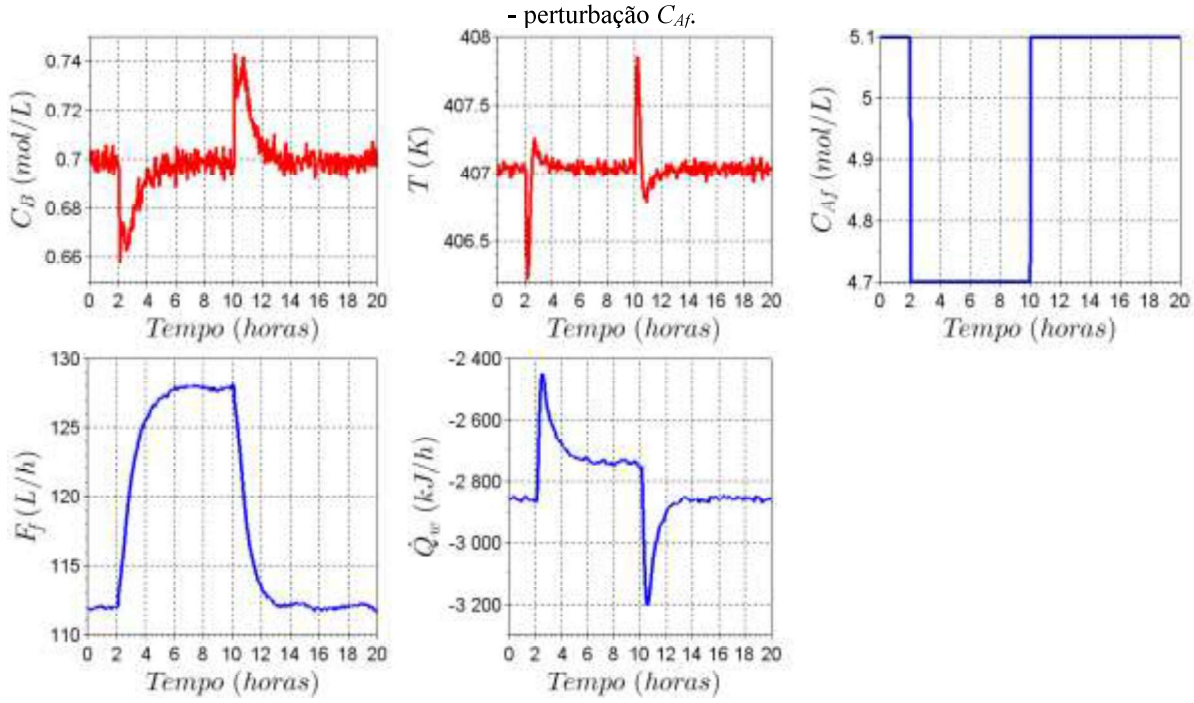
Os controles foram sintonizados através do método de ajuste de Ziegler-Nichols, sendo obtidos os seguintes parâmetros: para o controlador discreto PI de concentração C_B, foram obtidos k_{C1} = 8,0 e τ_{I1} = 0,5; para o controlador discreto PI de temperatura T, forma obtidos k_{C2} = 10,0 e τ_{I2} = 0,1. O tempo de amostragem da simulação foi de 0,05 h (3 min). O

emparelhamento realizado entre as variáveis manipuladas e as variáveis controladas foram F com C_B e Q_w com T.

A simulação do reator para produção de ciclopentanol foi implementada no *software* livre Scilab® 5.5.2. Para ilustrar os métodos, foram considerados 4 cenários de falhas, e através de simulação, foram obtidas as respostas das variáveis controladas através das variáveis manipuladas para cada uma dessas falhas.

A primeira falha, nomeada Perturbação C_{Af}, é uma perturbação na carga do sistema simulada como uma entrada degrau negativa na alimentação de reagente A, C_{Af}, que inicialmente possui um valor de 5,1 mol/L, e após a perturbação torna-se 4,7 mol/L. Apesar de denominarmos a perturbação em C_{Af} como uma falha de sistema, na realidade a operação reflete uma variação não prevista na alimentação do processo. Foi realizado o treinamento das metodologias de detecção de falhas de forma a reconhecer essa variação momentânea da corrente de alimentação como uma falha. A Figura 3.2 apresenta o comportamento das variáveis controladas (C_B e T) e manipuladas (F_f e $\dot{Q}_w$) sob a ação do controle regulador PI quando a Perturbação C_{Af} é aplicada. A perturbação foi aplicada no instante 2 h, e retirada no instante 10 h. Através da Figura 3.2, é possível verificar que os controladores PI foram capazes de rejeitar a perturbação, fazendo com que as variáveis controladas C_B e T retornassem a condição de *setpoint*.

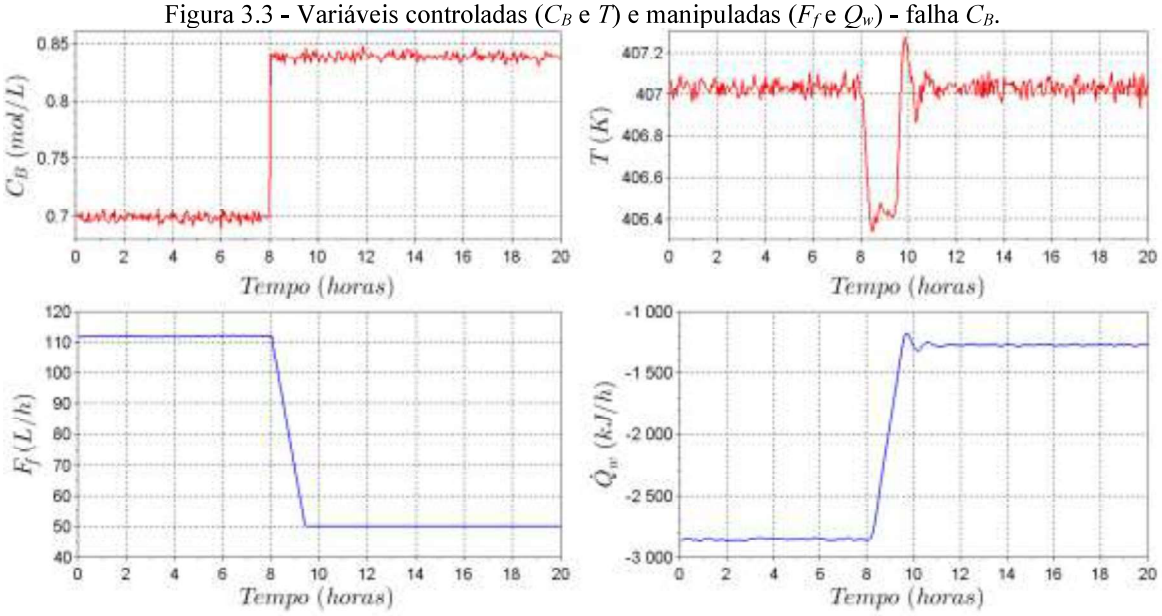
Figura 3.2 - Variáveis controladas (C_B e T), concentração de alimentação (C_{Af}) e variáveis manipuladas (F_f e Q_w) - perturbação C_{Af}.



A segunda falha, nomeada Falha C_B, é um aumento abrupto de 20% no valor medido da

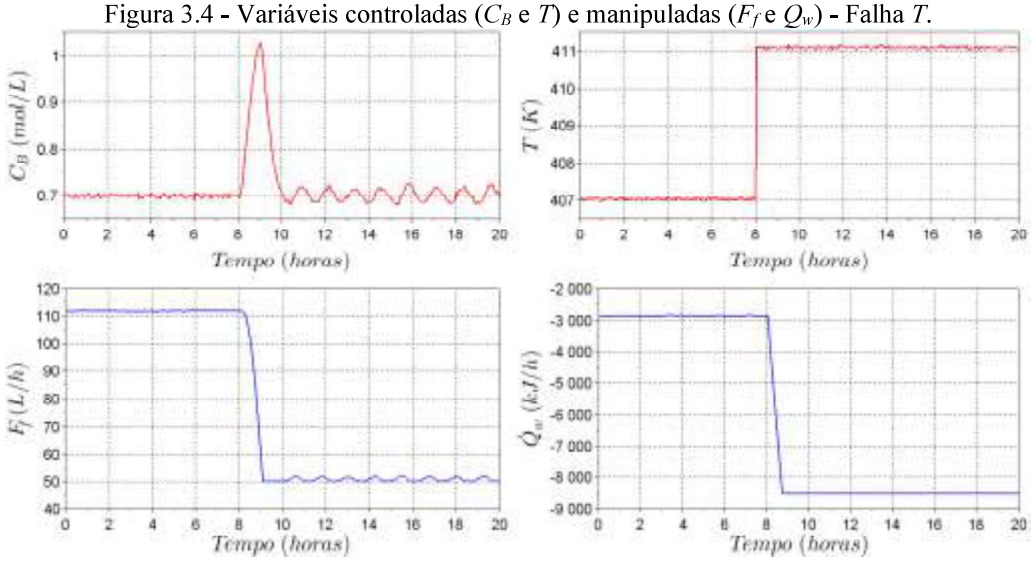
variável controlada C_B a partir do momento em que a falha foi aplicada, a partir do instante de 8 h, acrescida de um ruído aleatório gerado por distribuição gaussiana com média 0 e variância 10^{-5}. A Figura 3.3 apresenta o comportamento das variáveis controladas (C_B e T) e manipuladas (F_f e $\dot{Q}_w$) para a Falha C_B.

Figura 3.3 - Variáveis controladas (C_B e T) e manipuladas (F_f e Q_w) - falha C_B.



A terceira falha, nomeada Falha T, considera que o sensor de temperatura do reator, T, danifica-se a partir do instante 8 h, produzindo uma medida incorreta. A medida incorreta do sensor de temperatura é simulada como sendo um valor 1% maior que a última medida correta produzida pelo sensor antes de aplicada a falha. Todos os dados simulados são acrescidos de um ruído aleatório gerado por distribuição gaussiana com média 0 e variância 10^{-5} (para as concentrações) e 10^{-3} (para as outras variáveis). A Figura 3.4 mostra que a medida incorreta do sensor de temperatura que é enviada para o controlador PI faz com que ambas as variáveis controladas (F_f e $\dot{Q}_w$) passem a operar em seus limites extremos inferiores (50 L/h e -8500 kJ/h).

Figura 3.4 - Variáveis controladas (C_B e T) e manipuladas (F_f e Q_w) - Falha T.

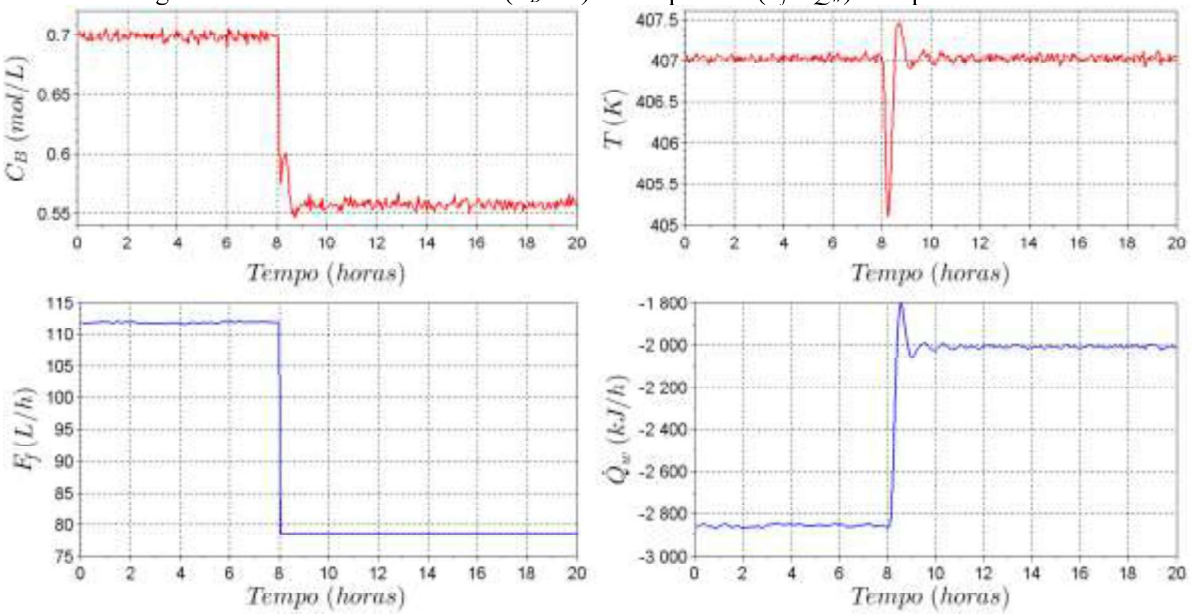


O comportamento dos sinais apresenta o acoplamento das variáveis controladas e a oscilação da concentração de ciclopentanol. Pode-se verificar que devido a modificação na temperatura T, a ação do controlador fez com que ambas as variáveis manipuladas, F_f e Q_w saturaram nos valores inferiores dos limites de ambas as variáveis.

A quarta e última falha simulada, nomeada Emperramento F_f, é o emperramento da válvula de alimentação do reagente A em uma posição que resulta em uma vazão 30% menor que a vazão de estado estacionário do processo. A Figura 3.5 apresenta o comportamento das variáveis controladas e manipuladas para o Emperramento F_f, aplicada no instante 8 h.

Juntamente com a operação normal do reator para a produção de ciclopentanol (Figura 3.6), operando em torno de um ponto estacionário especificado para as variáveis controladas (C_B e T) e manipuladas (F_f e $\dot{Q}_w$), utiliza-se todos os dados das falhas e operação normal para o treinamento das metodologias de detecção e diagnóstico de falhas. O ruído da variável controlada C_B foi estabelecido através de uma distribuição gaussiana aleatória com média 0 e variância 10^{-5}, somado ao valor do estado estacionário. Para a variável controlada T e as variáveis manipuladas F_f e $\dot{Q}_w$, foram adicionados a cada uma das variáveis em estado estacionário ruídos gaussianos aleatórios com média 0 e variância 10^{-3}. As oscilações presentes nos sinais são decorrentes da dinâmica do processo e do ruído adicionado.

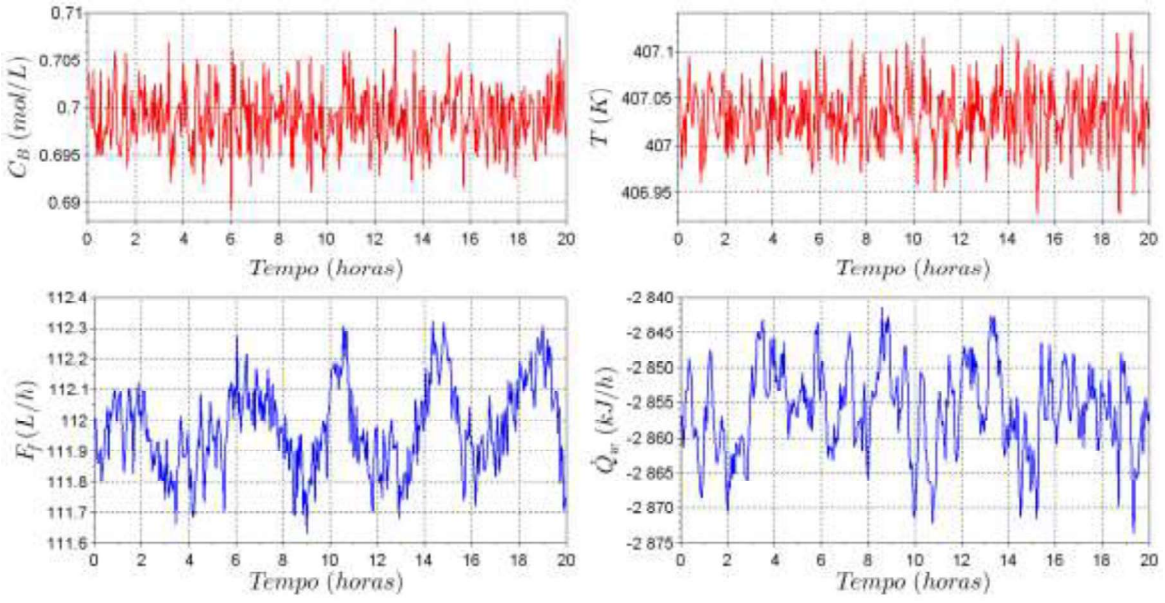
Figura 3.5 - Variáveis controladas (C_B e T) e manipuladas (F_f e Q_w) - Emperramento F.



3.5.2. Falhas com Diferentes Amplitudes

Para verificar a adaptabilidade dos modelos de detecção e diagnóstico de falhas, foram realizados testes dos modelos treinados com os dados apresentados nas Figuras Figura 3.2 a Figura 3.6 com dados amostrais em que a amplitude das falhas foram modificadas. Foram estabelecidos amplitudes maiores e menores para as falhas simuladas.

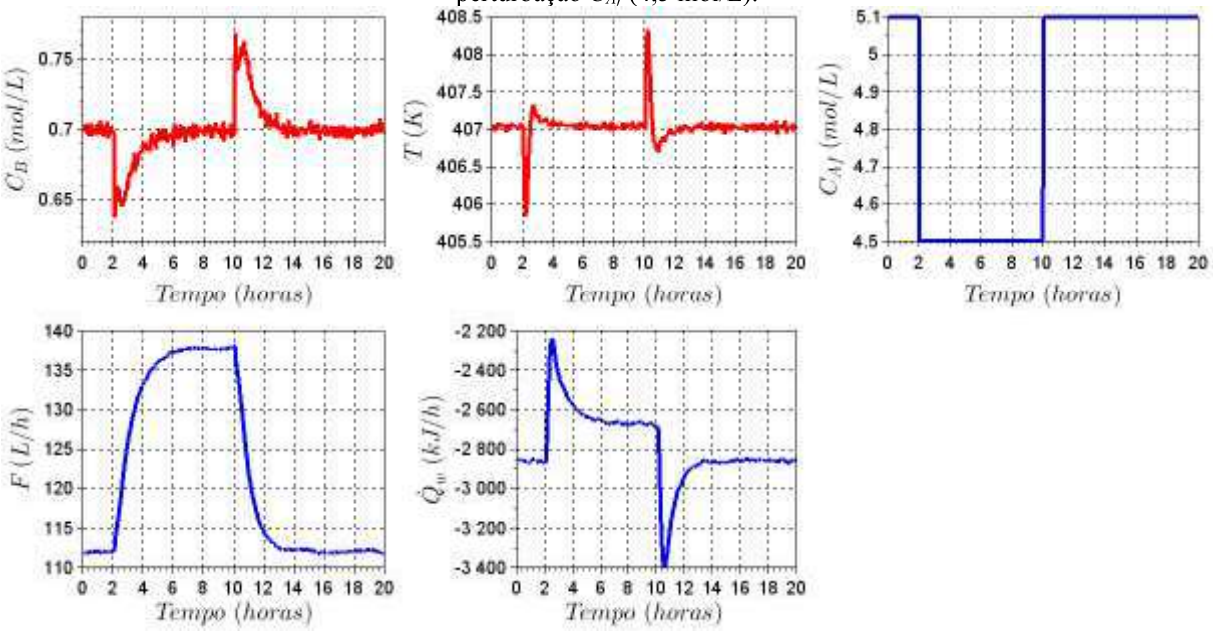
Figura 3.6 - Variáveis controladas (C_B e T) e manipuladas (F_f e Q_w) - Operação normal.



O estudo da diferença de amplitude da Falha 1, a Perturbação C_{Af}, foi realizado estabelecendo valores de C_{Af} menor (4,5 mol/L) e maior (4,9 mol/L) que os dados treinados (4,7 mol/L).

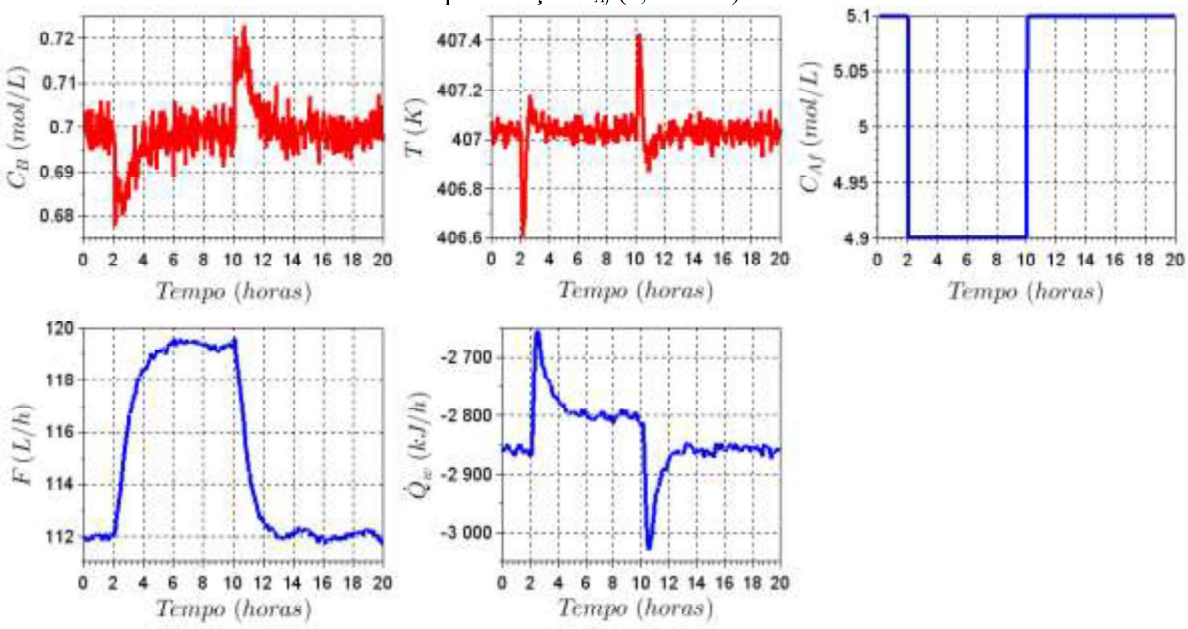
A Figura 3.7 apresenta o comportamento das variáveis com a perturbação em C_{Af}, de 5,1 mol/L para 4,5 mol/L, no intervalo de tempo de 2 h até 10 h.

Figura 3.7 - Variáveis controladas (C_B e T), manipuladas (F_f e Q_w) e concentração de alimentação (C_{Af}) - perturbação C_{Af} (4,5 mol/L).



A Figura 3.8 apresenta o comportamento das variáveis com a perturbação em C_{Af}, de 5,1 mol/L para 4,9 mol/L, no intervalo de tempo de 2 h até 10 h.

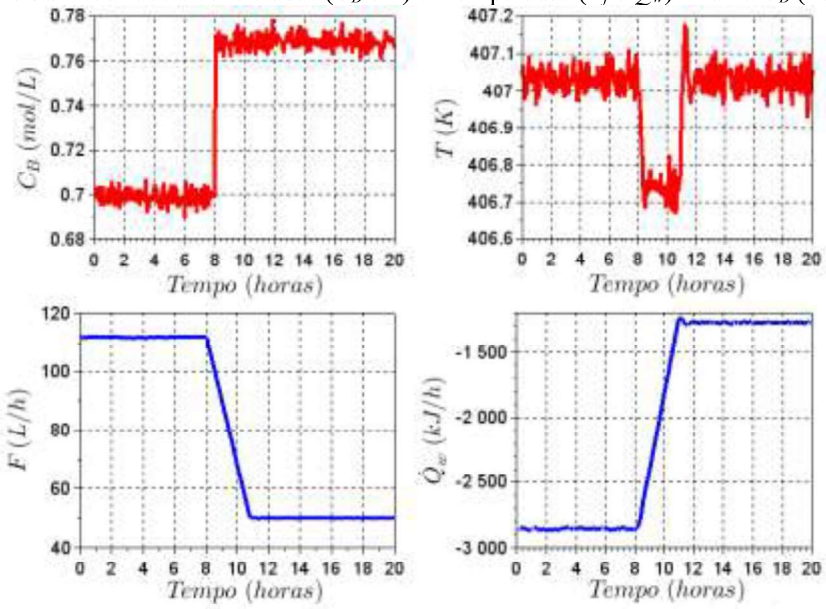
Figura 3.8 - Variáveis controladas (C_B e T), manipuladas (F_f e Q_w) e concentração de alimentação (C_{Af}) - perturbação C_{Af} (4,9 mol/L).



Para realizar o estudo da diferença de amplitude da Falha 2, a Falha C_B, foram estabelecidos aumentos abruptos menor (10%) e maior (30%) no valor medido da variável

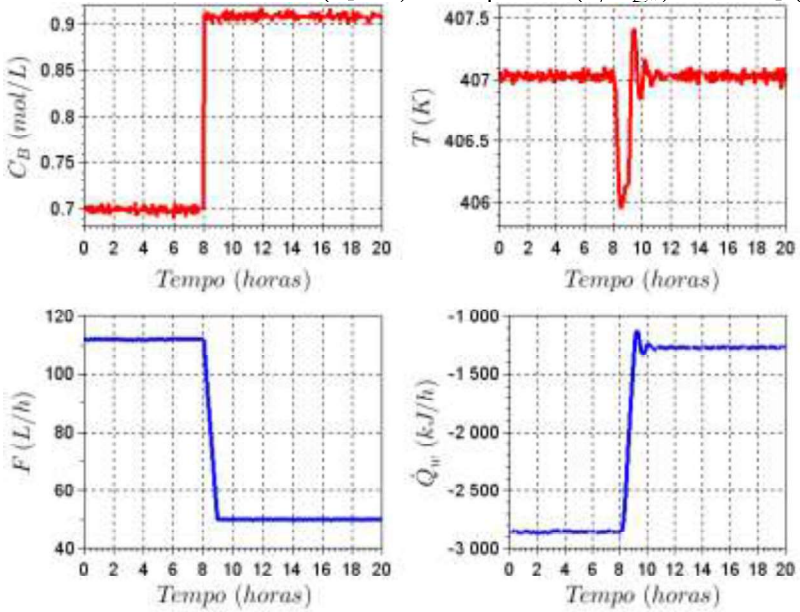
controlada C_B, diferentes do aumento abrupto treinado (20%), a partir do momento em que a falha foi aplicada no instante de 8 h. A Figura 3.9 apresenta o comportamento das variáveis controladas e manipuladas devido ao aumento abrupto de 10% no valor medido da variável controlada C_B após o instante 8 h.

Figura 3.9 - Variáveis controladas (C_B e T) e manipuladas (F_f e Q_w) - Falha C_B (10%).



A Figura 3.10 apresenta o comportamento das variáveis controladas e manipuladas devido ao aumento abrupto de 10% no valor medido da variável controlada C_B após o instante 8 h.

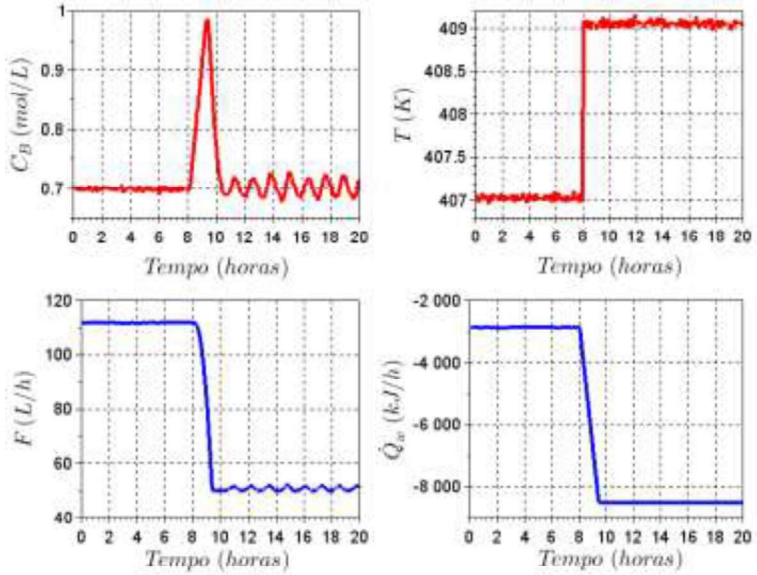
Figura 3.10 - Variáveis controladas (C_B e T) e manipuladas (F_f e Q_w) - Falha C_B (30%).



O estudo da modificação de amplitude da terceira falha, a Falha T, foi realizado através de falhas no sensor de temperatura do reator produzindo medidas incorretas da leitura do sensor

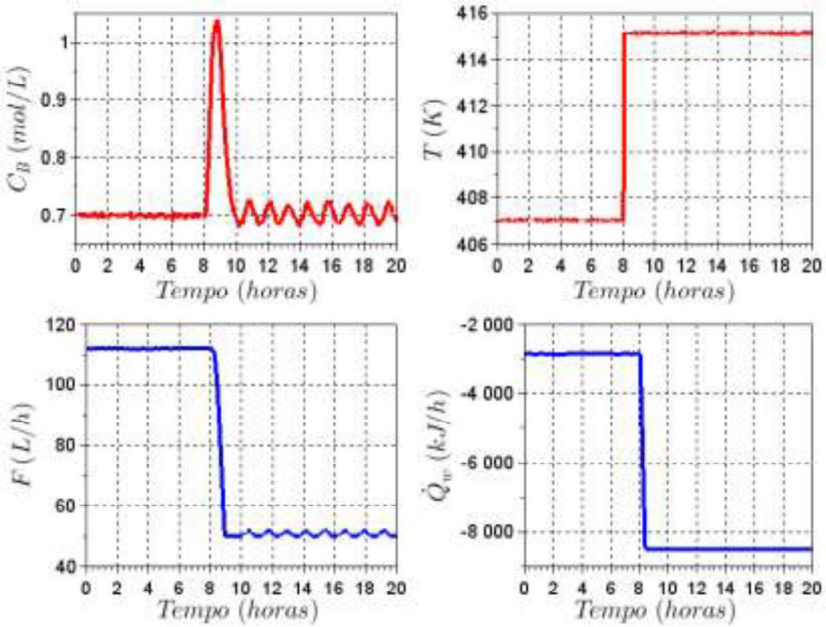
de temperatura do reator, estabelecendo medidas 0,5% maior e 2% maior que a última medida correta produzida pelo sensor antes de aplicada a falha, diferente dos dados treinados com 1% maior que a última medida correta produzida pelo sensor. A Figura 3.11 mostra a medida incorreta 0,5% maior do sensor de temperatura, saturando o controlador PI estabelecido.

Figura 3.11 - Variáveis controladas (C_B e T) e manipuladas (F_f e Q_w) - Falha T (0,5%).



A Figura 3.12 apresenta o comportamento das variáveis controladas e manipuladas após a medida incorreta 2% maior do sensor de temperatura, saturando o controlador PI estabelecido.

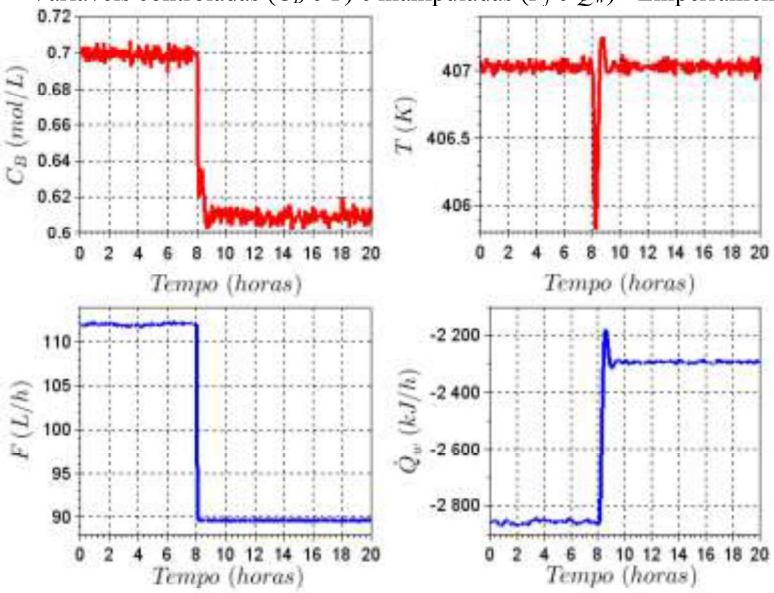
Figura 3.12 - Variáveis controladas (C_B e T) e manipuladas (F_f e Q_w) - Falha T (2%).



Para estudar a diferença de amplitude da Falha 4, Emperramento F, foram realizadas simulações em que a válvula de alimentação do reagente A emperra em posições que produzem vazões 20% e 40% menores que a do estado estacionário, valores menores e maiores que a falha

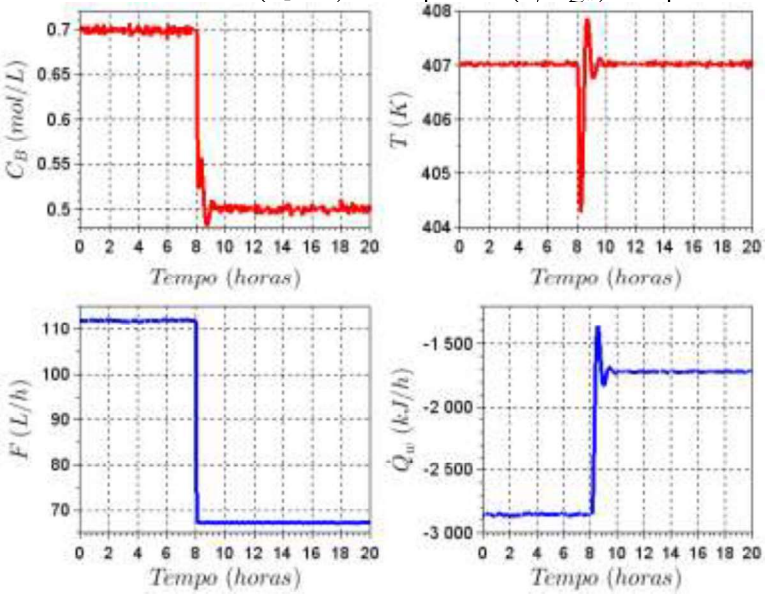
treinada, com posição que produz vazão 30% menor que a do estado estacionário. A Figura 3.14 apresenta o comportamento das variáveis controladas e manipuladas para o Emperramento F_f, com posição 20% menor da abertura de estado estacionário, aplicada no instante 8 h.

Figura 3.13 - Variáveis controladas (C_B e T) e manipuladas (F_f e Q_w) - Emperramento F (20%).



A Figura 3.14 apresenta o comportamento das variáveis controladas e manipuladas para o Emperramento F_f, com posição 40% menor da abertura de estado estacionário, aplicada no instante 8 h.

Figura 3.14 - Variáveis controladas (C_B e T) e manipuladas (F_f e Q_w) - Emperramento F (40%).



3.5.3. Saída do Processo de Detecção

Além dos dados normais e de operações com falhas, foram estabelecidos rótulos para cada dado amostral, que representam as diferentes operações que o sistema está passando. Os

rótulos são utilizados para os sistemas de detecção e diagnóstico de falhas nas etapas de treinamento e validação, como saídas desses sistemas. Foram estabelecidos rótulos para a Operação Normal, a Perturbação C_{Af}, a Falha C_B, a Falha T e o Emperramento F_f como 1, 2, 3, 4 e 5, respectivamente. O rótulo é designado ao dado amostral em todos os instantes dos conjuntos de dados de treinamento e validação para cada uma das operações (Equação (3.11)).

$$y_i = [1, 5], i \in \Re, \tag{3.11}$$

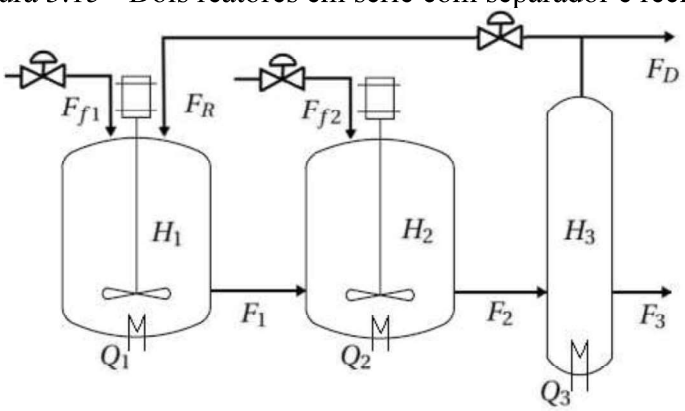
O vetor de rótulos é utilizado como saída dos métodos de detecção e diagnóstico de falhas, principalmente nos métodos com aprendizagem supervisionada.

No próximo tópico, é apresentado o segundo estudo de caso utilizado para demonstrar as qualidades e defeitos dos diferentes métodos de detecção e diagnóstico, além de outro uso prático das técnicas estudadas neste trabalho posteriormente.

3.6.Estudo de Caso 2 - Planta Química

Neste segundo estudo de caso, considera-se a planta apresentada na Figura 3.15 e descrita em Stewart e colaboradores (2011). A planta química é formada por dois reatores perfeitamente agitados (CSTR, *Continuous Stirred-tank Reactor*) em série, em que correntes de reagente A entram nos reatores e o mesmo reagente A é transformado no produto B através de uma reação de primeira ordem. O produto desejado B é consumido através de uma reação paralela de primeira ordem que forma o subproduto C. Após os dois reatores CSTR, um separador tipo *flash* destila o subproduto, fazendo com que parte deste seja redirecionado ao primeiro reator.

Figura 3.15 - Dois reatores em série com separador e reciclo.



Fonte: adaptado de STEWART *et al*., 2011.

O modelo foi adaptado de Stewart *et al*. (2011), resultando nas seguintes equações:

CSTR-1:

$$\frac{dH_1}{dt} = \frac{1}{\rho A_1}\left(F_{f1} + F_R - F_1\right) \tag{3.12}$$

$$\frac{dx_{A1}}{dt} = \frac{1}{\rho A_1 H_1}\left[F_{f1}(x_{A0} - x_{A1}) + F_R(x_{AR} - x_{A1})\right] - k_{A1}x_{A1} \tag{3.13}$$

$$\frac{dx_{B1}}{dt} = \frac{1}{\rho A_1 H_1}\left[F_{f1}(x_{B0} - x_{B1}) + F_R(x_{BR} - x_{B1})\right] + k_{A1}x_{A1} - k_{B1}x_{B1} \tag{3.14}$$

$$\frac{dT_1}{dt} = \frac{1}{\rho A_1 H_1}\left[F_{f1}(T_0 - T_1) + F_R(T_R - T_1)\right] + \frac{Q_1}{\rho A_1 c_P H_1} - \frac{1}{c_P}(k_{A1}x_{A1}\Delta H_A + k_{B1}x_{B1}\Delta H_B) \tag{3.15}$$

CSTR-2:

$$\frac{dH_2}{dt} = \frac{1}{\rho A_2}\left(F_{f2} + F_1 - F_2\right) \tag{3.16}$$

$$\frac{dx_{A2}}{dt} = \frac{1}{\rho A_2 H_2}\left[F_{f2}(x_{A0} - x_{A2}) + F_1(x_{A1} - x_{A2})\right] - k_{A2}x_{A2} \tag{3.17}$$

$$\frac{dx_{B2}}{dt} = \frac{1}{\rho A_2 H_2}\left[F_{f2}(x_{B0} - x_{B2}) + F_1(x_{B1} - x_{B2})\right] + k_{A2}x_{A2} - k_{B2}x_{B2} \tag{3.18}$$

$$\frac{dT_2}{dt} = \frac{1}{\rho A_2 H_2}\left[F_{f2}(T_0 - T_2) + F_1(T_1 - T_2)\right] + \frac{Q_2}{\rho A_2 c_P H_2} - \frac{1}{c_P}(k_{A2}x_{A2}\Delta H_A + k_{B2}x_{B2}\Delta H_B) \tag{3.19}$$

SEPARADOR:

$$\frac{dH_3}{dt} = \frac{1}{\rho A_3}(F_2 - F_D - F_R - F_3) \tag{3.20}$$

$$\frac{dx_{A3}}{dt} = \frac{1}{\rho A_3 H_3}[F_2(x_{A2} - x_{A3}) - F_D(x_{A3} - x_{AR}) - F_R(x_{A3} - x_{AR})] \tag{3.21}$$

$$\frac{dx_{B3}}{dt} = \frac{1}{\rho A_3 H_3}[F_2(x_{B2} - x_{B3}) - F_D(x_{B3} - x_{BR}) - F_R(x_{B3} - x_{BR})] \tag{3.22}$$

$$\frac{dT_3}{dt} = \frac{1}{\rho A_3 H_3}[F_2(T_2 - T_3) - F_D(T_3 - T_R) - F_R(T_3 - T_R)] + \frac{Q_3}{\rho A_3 c_P H_3} \tag{3.23}$$

As vazões são definidas como:

$$F_i = k_{vi} H_i \tag{3.24}$$

e as constantes de reação:

$$k_{Ai} = k_A \exp\left(-\frac{E_A}{RT_i}\right) \tag{3.25}$$

$$k_{Bi} = k_B \exp\left(-\frac{E_B}{RT_i}\right) \tag{3.26}$$

em que para qualquer $i \in I_{1:3}$. A vazão de reciclo e os percentuais satisfazem as seguintes equações:

$$F_D = 0{,}01 F_R \tag{3.27}$$

$$x_{AR} = \frac{\alpha_A x_{A3}}{\bar{x}_3} \tag{3.28}$$

$$x_{BR} = \frac{\alpha_B x_{B3}}{\bar{x}_3} \tag{3.29}$$

$$\bar{x}_3 = \alpha_A x_{A3} + \alpha_B x_{B3} + \alpha_C x_{C3} \tag{3.30}$$

$$x_{C3} = (1 - x_{A3} - x_{B3}) \tag{3.31}$$

Os significados de cada variável utilizada no modelo podem ser encontrados na Tabela 3.2 e na Tabela 3.3 tem-se os parâmetros do modelo. A Fonte: STEWART *et al.*, 2011.

Tabela 3.4 mostra os estados estacionários do sistema considerados neste trabalho.

Tabela 3.2 - Variáveis do processo do estudo de caso da planta química.

H_1, H_2, H_3	Alturas de líquido em cada equipamento
x_{A1}, x_{B1}, x_{C1}	Composições percentuais de *A*, *B* e *C* no CSTR-1
x_{A2}, x_{B2}, x_{C2}	Composições percentuais de *A*, *B* e *C* no CSTR-2
x_{A3}, x_{B3}, x_{C3}	Composições percentuais de *A*, *B* e *C* na saída inferior do separador
x_{AR}, x_{BR}, x_{CR}	Composições percentuais de *A*, *B* e *C* na saída superior do separador
x_{A0}, x_{B0}	Composições percentuais de *A* e *B* nas correntes de alimentação
F_{f1}, F_{f2}	Correntes de alimentação de *A* puro nos CSTR-1 e CSTR-2
T_0	Temperatura das correntes de alimentação
T_R	Temperatura da corrente de reciclo
F_1, F_2, F_3	Correntes efluentes em cada equipamento

F_D	Corrente efluente do processo
F_R	Corrente de reciclo do separador para o CSTR-1
ρ	Densidade da mistura reacional
c_P	Calor específico da mistura reacional
ΔH_A, ΔH_B	Calores das reações
A_1, A_2, A_3	Áreas das seções transversais de cada equipamento
T_1, T_2, T_3	Temperaturas em cada equipamento
Q_1, Q_2, Q_3	Entradas externas de calor/refrigerante para cada equipamento
k_{A1}, k_{B1}	Constantes de reação no CSTR-1
k_{A2}, k_{B2}	Constantes de reação no CSTR-2

Tabela 3.3 - Parâmetros do modelo da planta química.

Parâmetros	
$A_1 = 3$ m²	$k_A = 2$ 1/min
$A_2 = 3$ m²	$k_B = 0{,}18$ 1/min
$A_3 = 1$ m²	$E_A/R = 300$ K
$\rho = 1000$ kg/m³	$E_B/R = 300$ K
$C_p = 5$ kJ/(kg.K)	$\Delta H_A = -200$ kJ/kg
$k_{v1} = 1020$ kg/(m.min)	$\Delta H_B = -50$ kJ/kg
$k_{v2} = 1020$ kg/(m.min)	$\alpha_A = 1{,}5$ [-]
$k_{v3} = 1020$ kg/(m.min)	$\alpha_B = 1{,}5$ [-]
$x_{A0} = 1$ %	$\alpha_C = 2{,}0$ [-]
$x_{B0} = 0$ %	$T_0 = 313$ K

Fonte: STEWART *et al.*, 2011.

Tabela 3.4 - Estados estacionários da planta química.

1º Estado Estacionário ($ee1$)	**2º Estado Estacionário** ($ee2$)
$H_{1ee1} = 0{,}706$ m	$H_{1ee2} = 0{,}423$ m
$H_{2ee1} = 1{,}176$ m	$H_{2ee2} = 0{,}541$ m
$H_{3ee1} = 0{,}939$ m	$H_{3ee2} = 0{,}351$ m
$x_{A1ee1} = 0{,}210$	$x_{A1ee2} = 0{,}177$
$x_{A2ee1} = 0{,}150$	$x_{A2ee2} = 0{,}101$
$x_{A3ee1} = 0{,}200$	$x_{A3ee2} = 0{,}142$
$x_{B1ee1} = 0{,}626$	$x_{B1ee2} = 0{,}650$
$x_{B2ee1} = 0{,}597$	$x_{B2ee2} = 0{,}605$
$x_{B3ee1} = 0{,}796$	$x_{B3ee2} = 0{,}851$
$T_{1ee1} = 348{,}107$ K	$T_{1ee2} = 350{,}958$ K
$T_{2ee1} = 350{,}660$ K	$T_{2ee2} = 354{,}553$ K
$T_{3ee1} = 350{,}670$ K	$T_{3ee2} = 354{,}651$ K
$F_{f1ee1} = 480{,}000$ kg/min	$F_{f1ee2} = 240{,}000$ kg/min
$F_{f2ee1} = 480{,}000$ kg/min	$F_{f2ee2} = 120{,}000$ kg/min
$F_{Ree1} = 240{,}000$ kg/min	$F_{Ree2} = 192{,}000$ kg/min
$Q_{1ee1} = 60{,}000$ kJ/min	$Q_{1ee2} = 180{,}000$ kJ/min
$Q_{2ee1} = 60{,}000$ kJ/min	$Q_{2ee2} = 210{,}000$ kJ/min
$Q_{3ee1} = 60{,}000$ kJ/min	$Q_{3ee2} = 270{,}000$ kJ/min

As saídas controladas e as entradas manipuladas são indicadas, respectivamente, nas Equações (3.32) e (3.33).

$$\mathbf{y} = [H_1, \quad T_1, \quad H_2, \quad T_2, \quad H_3, \quad T_3]^T \tag{3.32}$$

$$\mathbf{u} = \left[F_{f1}, \quad Q_1, \quad F_{f2}, \quad Q_2, \quad F_R, \quad Q_3\right]^T \tag{3.33}$$

No presente sistema, foi utilizado um controle descentralizado nos estados estacionários *ee*1 e *ee*2, cuja formulação se encontra em Rocha (2018).

3.6.1. Características do Processo - Planta Química

Os parâmetros de sintonia dos controladores preditivos baseados em modelo (MPC, *Model Predictive Controller*) utilizados para a planta química, tem-se as seguintes escolhas para as matrizes de ponderação e horizontes de predição e de controle:

$$\mathbf{Q} = 10 \cdot \mathbf{I}_{n_y \times n_y} \tag{3.34}$$

$$\mathbf{R} = \mathbf{I}_{m \times m} \tag{3.35}$$

$$\mathbf{W} = \mathbf{I}_{m \times m} \tag{3.36}$$

$$H_p = 5 \tag{3.37}$$

$$H_u = 5 \tag{3.38}$$

em que n_y representa o número de saídas controladas; m o número de entradas manipuladas e a matriz **I** representa a matriz identidade.

O tempo de amostragem escolhido nesse estudo de caso foi igual a $T_s = 0{,}5$ min. Essa escolha baseou-se na dinâmica mais veloz, entre todas as dinâmicas das variáveis envolvidas quando a operação da planta encontra-se em malha aberta.

Foram construídas trajetórias de referência para as variáveis controladas para obtenção dos dados de treinamento para os diferentes sistemas de detecção e classificação de falhas, utilizadas aqui neste estudo de caso para reconhecer os diferentes estados estacionários e as falhas simuladas. Para cada método, foram estabelecidos probabilidades e/ou rótulos de acordo com cada estado estacionário definido na trajetória de referência, além de cada falha simulada. Após a obtenção dos dados simulados das variáveis manipuladas e controladas do sistema de controle em situação normal e em falha, foram utilizadas somente as variáveis controladas como entrada dos sistemas de detecção. As saídas do sistema de detecção são os rótulos/probabilidades, dependendo do método de detecção supervisionado utilizado.

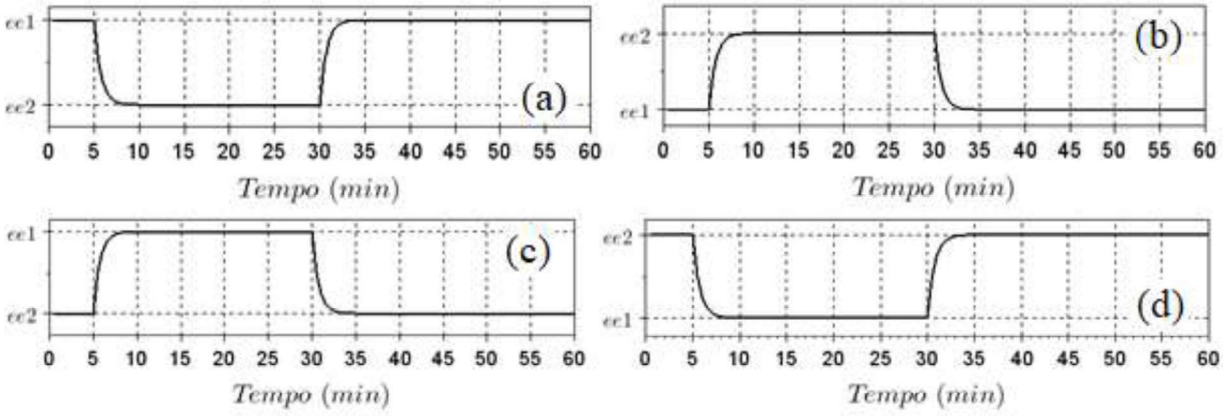
Foram aplicadas falhas nas variáveis manipuladas e controladas, de acordo com a Tabela 3.5.

Tabela 3.5 - Situações aplicadas a planta química.

Estado	Amplitude	Rótulo	Estado	Amplitude	Rótulo
Q_3 (ee1)	0,70	1	ee2	-	15
F_R (ee1)	1,30	2	H_1 (ee2)	0,70	16
T_3 (ee1)	0,98	3	T_1 (ee2)	1,02	17
H_3 (ee1)	1,30	4	F_{f1} (ee2)	0,70	18
Q_2 (ee1)	070	5	Q_1 (ee2)	1,30	19
F_{f2} (ee1)	1,30	6	H_2 (ee2)	0,70	20
T_2 (ee1)	0,98	7	T_2 (ee2)	1,02	21
H_2 (ee1)	1,30	8	Ff_2 (ee2)	0,70	22
Q_1 (ee1)	0,70	9	Q_2 (ee2)	1,30	23
F_{f1} (ee1)	1,30	10	H_3 (ee2)	0,70	24
T_1 (ee1)	0,98	11	T_3 (ee2)	1,02	25
H_1 (ee1)	1,30	12	F_R (ee2)	0,70	26
ee1	-	13	Q_3 (ee2)	1,30	27
0	-	14	-	-	-

Para o treinamento dos sistemas de detecção, foram simulados sinais de acordo com a trajetória de referência, em que se aplicaram as diferentes falhas após o instante 30 min, ao invés da modificação de estado estacionário. Os dados de treinamento são todos os sinais simulados com a trajetória de referência em situação normal (sem falha – 2 estados estacionários) e em cada uma das falhas (24 diferentes falhas, sendo 12 falhas em cada estado estacionário). A Figura 3.16 apresenta as trajetórias de referência de operação normal utilizadas na simulação dos dados de treinamento.

Figura 3.16 - Referenciais (operação normal) utilizados para os dados de treinamento.



(a) e (b): Transição *ee*1-*ee*2-*ee*1; (c) e (d): Transição *ee*2-*ee*1-*ee*2.

Para os referenciais em situações normais e com as falhas apresentadas na Tabela 3.5, foram simulados os dados das variáveis controladas e aplicadas as falhas respectivas, após o instante 30 minutos.

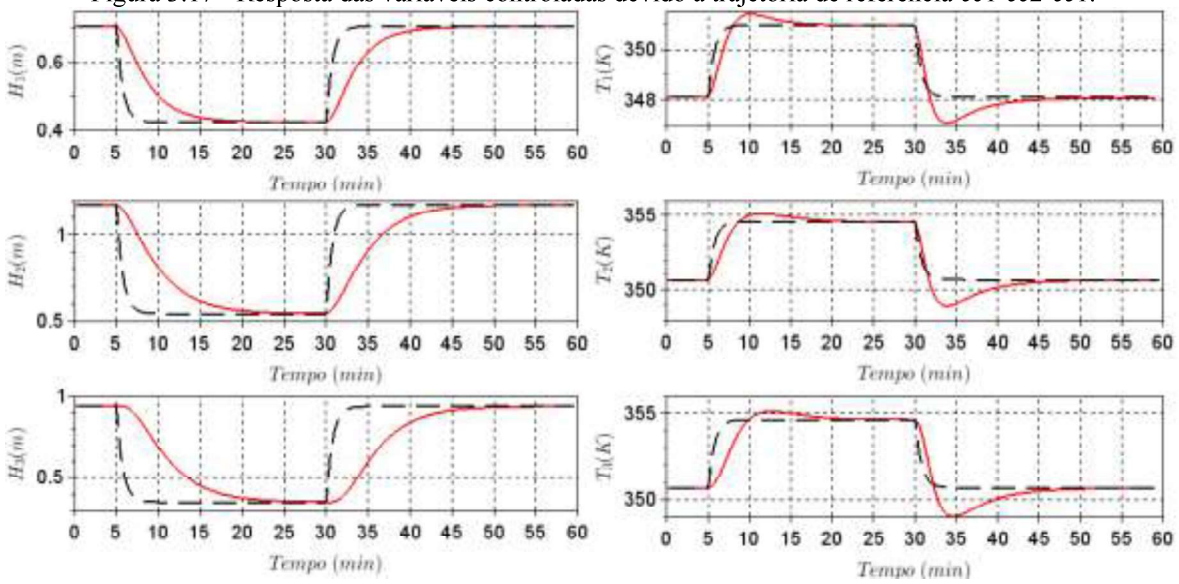
O referencial foi escolhido para estabelecer situações que apresentam dados de treinamento com transição do primeiro estado estacionário (*ee*1) para o segundo estado

estacionário (*ee*2), quanto situações que apresentam dados de treinamento com transição do segundo estado estacionário (*ee*2) para o primeiro estado estacionário (*ee*1).

As trajetórias de referência foram construídas de forma que não apresentassem prioridade sobre a transição entre estados estacionários ou sobre os períodos em que o sistema estivesse em repouso.

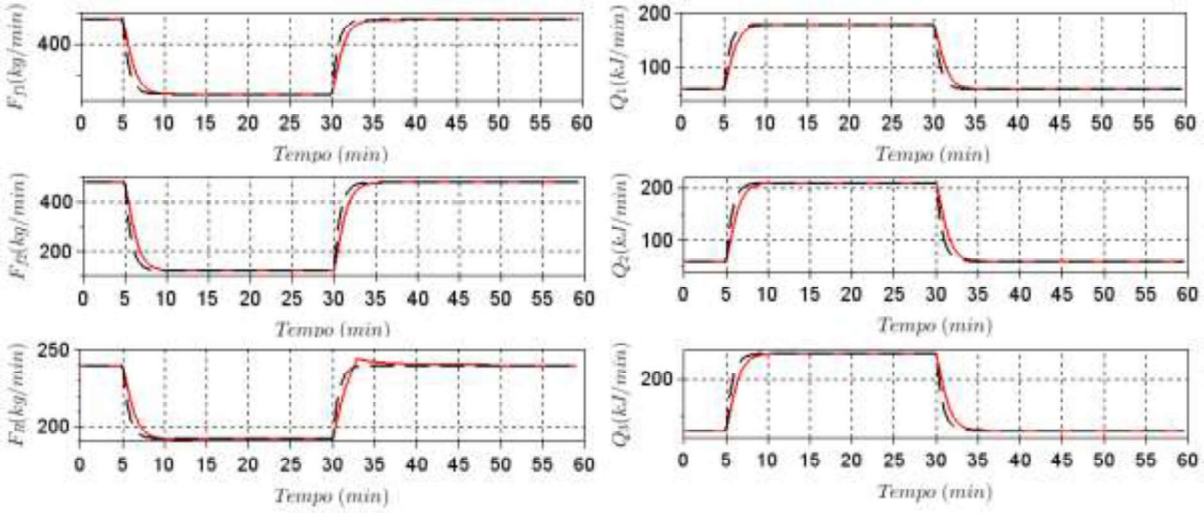
A Figura 3.17 apresenta os comportamentos das variáveis controladas (Equação (3.32)) para a trajetória de referência *ee*1-*ee*2-*ee*1.

Figura 3.17 - Resposta das variáveis controladas devido a trajetória de referência *ee*1-*ee*2-*ee*1.



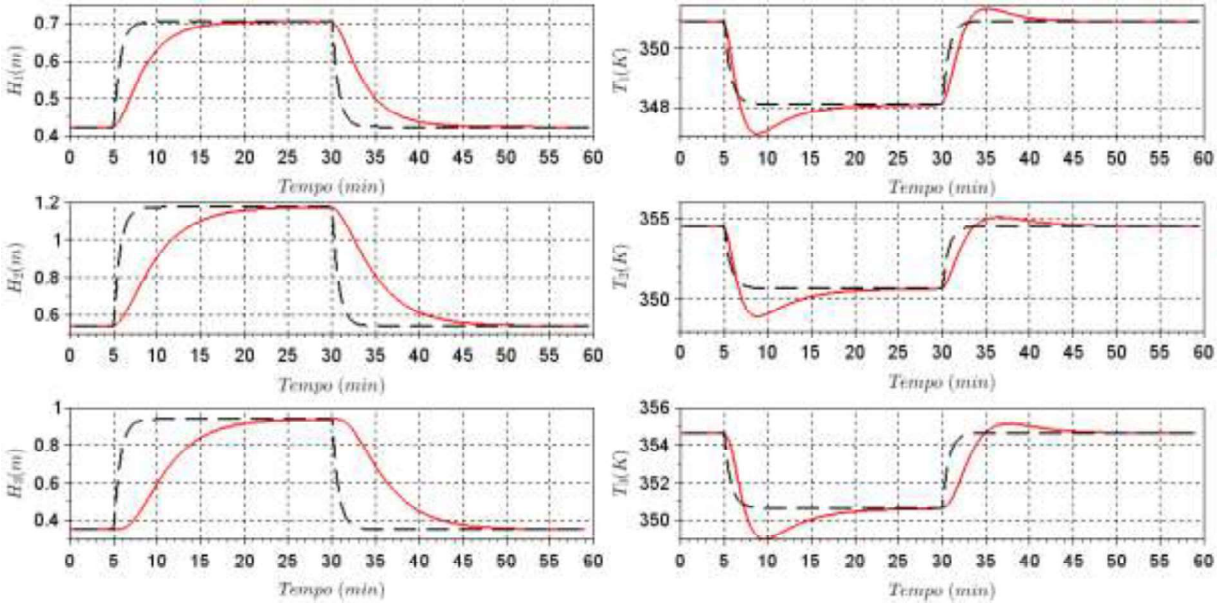
A Figura 3.18 apresenta os comportamentos das variáveis manipuladas (Equação (3.33) para a trajetória de referência *ee*1-*ee*2-*ee*1.

Figura 3.18 - Resposta das variáveis manipuladas devido a trajetória de referência *ee*1-*ee*2-*ee*1.



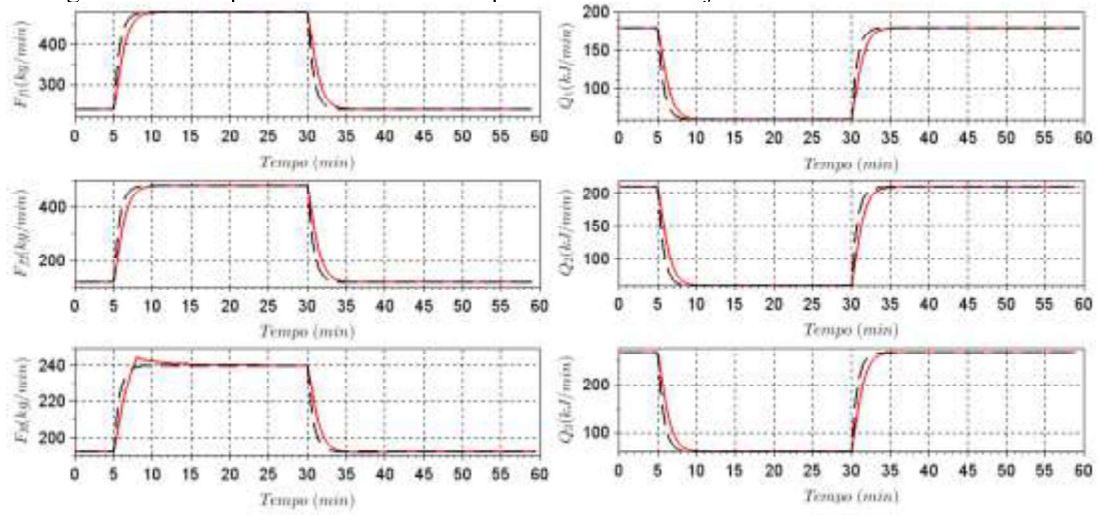
A Figura 3.19 apresenta os comportamentos das variáveis controladas (Equação (3.32)) para a trajetória de referência *ee*2-*ee*1-*ee*2.

Figura 3.19 - Resposta das variáveis controladas devido a trajetória de referência *ee*2-*ee*1-*ee*2.



A Figura 3.20 apresenta os comportamentos das variáveis manipuladas (Equação (3.33) para a trajetória de referência *ee*2-*ee*1-*ee*2.

Figura 3.20 - Resposta das variáveis manipuladas devido a trajetória de referência *ee*2-*ee*1-*ee*2.



Para cada falha simulada utilizada no treinamento dos sistemas de detecção de falhas, foram aproveitados os referenciais acima, modificados a partir do instante 30 minutos, em que foram aplicadas as falhas estabelecidas na Tabela 3.5. A compilação dos dados utilizados para o treinamento dos sistemas de detecção de falhas está apresentada no Apêndice A.

3.6.2. Saída do Processo de Treinamento

Além dos dados simulados das variáveis controladas para a trajetória de referência construída, foram estabelecidos rótulos para cada dado amostral, que representam os diferentes estados estacionários em que o sistema passa, além das diferentes falhas. Os rótulos são utilizados para os sistemas de detecção e diagnóstico nas etapas de treinamento e validação, como saídas desses sistemas. Foram estabelecidos rótulos para o Estado Estacionário 1 ($ee1$), para o Estado Estacionário 2 ($ee2$) como 13 e 15, respectivamente, e para cada falha de acordo com a Tabela 3.5. O rótulo é designado ao dado amostral em todos os instantes dos conjuntos de dados de treinamento e validação para cada uma das operações (Equação (3.39)).

$$y_i = [1, 27], i \in \Re, \tag{3.39}$$

O vetor de rótulos foi utilizado como saída dos métodos de detecção e diagnóstico, principalmente nos métodos com aprendizagem supervisionada.

3.6.3. Detecção

A simulação da planta química foi implementada no *software* livre Scilab® 5.5.2. Para ilustrar os métodos, foram considerados 26 cenários diferentes dos dados utilizados em treinamento em que ocorriam transições entre os estados estacionários 1 e 2, que partem tanto do estado estacionário 1 para o 2, quanto do 2 para 1, além de falhas ocorridas quando o sistema se encontrava no estado estacionário 2 ou no estado estacionário 1, respectivamente. As variáveis controladas foram obtidas a partir da simulação das variáveis manipuladas dentro das transições e aplicações das falhas. Todos esses cenários estão apresentados no Apêndice A.

No próximo capítulo discorre-se acerca da teoria sobre máquinas de vetores de suporte de classificação e de regressão, apresentando as metodologias de detecção e diagnóstico de falhas baseadas em SVM. Os resultados obtidos da aplicação dos métodos de detecção e diagnóstico de falhas para os estudos de casos foram inseridos ao final dos capítulos seguintes.

CAPÍTULO 4
DETECÇÃO DE FALHAS COM MÁQUINAS DE VETORES DE SUPORTE

4.1. Introdução

Este capítulo traz uma revisão da metodologia das máquinas de vetores de suporte (*Support Vector Machines* - SVM) dentro da detecção de falhas de controle. São apresentadas diversas características da metodologia SVM, da detecção de falhas utilizando a mesma, além dos resultados dos estudos de casos resolvidos com a metodologia em questão.

Antes de apresentar a metodologia SVM propriamente dita, faz-se necessário introduzir a teoria de aprendizado estatístico, abordada a seguir.

4.2. Teoria de Aprendizado Estatístico

Esta seção consiste em uma breve introdução à teoria de aprendizado estatístico. Para um maior aprofundamento sobre o assunto, vide Vapnik (1998). A Figura 4.1 apresenta os erros de modelagem.

Figura 4.1 - Erros de Modelagem.

O objetivo de uma modelagem é encontrar um modelo do espaço de hipótese, que seja próximo (a respeito de alguma medida de erro) à função subjacente no espaço alvo. Ao modelar, podem existir dois tipos de erros nas seguintes circunstâncias: erro de aproximação e erro de estimativa.

Denomina-se ***Erro de Aproximação*** (f_n) quando o espaço de hipótese é menor que o espaço alvo, e, portanto, a função subjacente pode estar fora do espaço de hipótese. Uma má escolha do espaço do modelo irá resultar em um grande erro de aproximação, que será apresentado como uma divergência do modelo

Denominados ***Erro de Estimativa*** ($\hat{f}_{n,N}$) quando no procedimento de aprendizado ocorre a seleção de um modelo não ótimo a partir do espaço de hipótese.

Juntos, os erros de aproximação e estimativa formam o que denomina-se o erro de generalização. Para correção do erro de generalização contido em um modelo, é necessário encontrar uma função f que minimize o risco R (Equação (4.1)),

$$R(f) = \int L(f(\mathbf{x}), y) dP(\mathbf{x}, y) \tag{4.1}$$

em que $L(f(\mathbf{x}),y)$ é o resultado da escolha da função de custo entre $f(\mathbf{x})$ e y. Uma modalidade de função de custo utilizada em classificação de dados é definida como (Equação (4.2)

$$L(f(\mathbf{x}), y) = \begin{cases} 0 \text{ se } y = f(\mathbf{x}) \\ 1 \text{ se } y \neq f(\mathbf{x}) \end{cases} \tag{4.2}$$

Entretanto, a distribuição de probabilidade conjunta $P(\mathbf{x}, y)$ é desconhecida, sendo impossível minimizar o risco diretamente. É possível encontrar uma aproximação de acordo com o princípio de minimização de risco empírico R_{emp} (Equação (4.3),

$$R_{emp}(f) = \frac{1}{l} \sum_{i=1}^{l} L(f(\mathbf{x}_i), y_i) \tag{4.3}$$

em que l é o número total de dados. O objetivo é encontrar a função $\hat{f}_{n,l}(\mathbf{x})$, dentre a classe de funções $f(\mathbf{x})$ que minimiza a função de risco $R_{emp}(f)$. A Equação (4.3 minimiza o risco empírico (Equação (4.4):

$$\hat{f}_{n,l}(\mathbf{x}) = \arg\min_{f \in S_n} R_{emp}(f) \tag{4.4}$$

A minimização do risco empírico somente faz sentido se (Equação (4.5)),

$$\lim_{l \to \infty} R_{emp}(f) = R(f) \tag{4.5}$$

o que é verdadeiro a partir da lei dos grandes números. No entanto, a minimização do risco empírico também deve satisfazer a Equação (4.6),

$$\lim_{l \to \infty} \min_{f \in S_n} R_{emp}(f) = \min_{f \in S_n} R(f) \tag{4.6}$$

que será válida quando o espaço de hipótese S_n for "pequeno" o suficiente. Esta condição é menos intuitiva e requer que os mínimos também convirjam. A teoria de convergência uniforme em probabilidade também proporciona um limitante para a variação do risco esperado $R(f)$ em função do risco empírico $R_{emp}(f)$. Um limitante típico que acontece com probabilidade $1-\delta$, tem a seguinte forma (Equação (4.7):

$$R(f) \leq R_{emp}(f) + \sqrt{\frac{h\ln\left(\frac{2l}{h}+1\right) - \ln\left(\frac{\delta}{4}\right)}{l}} \tag{4.7}$$

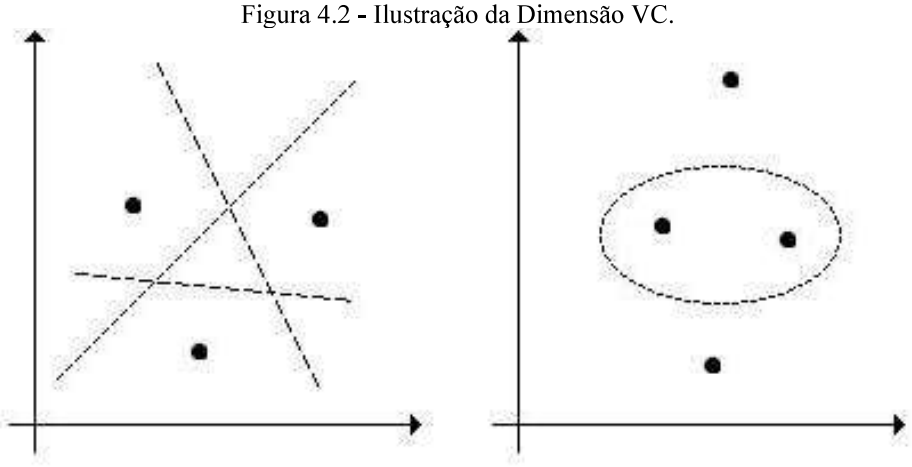
em que h é a dimensão VC.

Notavelmente, esta expressão para o risco esperado é independente da distribuição de probabilidades.

Dimensão VC:

A Dimensão VC é definida como a quantidade máxima de exemplos de treinamento (m) que podem ser classificados, sem erros, por um conjunto de funções (BURGES, 1998). Caso m não exista, a dimensão VC é definida como infinita (∞) (SCHÖLKOPF; SMOLA, 2002). A dimensão VC é utilizada para medir a capacidade do conjunto de funções $\mathfrak{F}$ do qual $\hat{f}$ é escolhida (Figura 4.2).

Figura 4.2 - Ilustração da Dimensão VC.



Definição 4.1 (Vapnik-Chervonenkis). A dimensão VC de um conjunto de funções é p se, e, somente se, existir um conjunto de pontos $\{x_i\}_{i=1}^{p}$ que possam ser separados em todas as 2^p possíveis maneiras, e não exista conjunto $\{x_i\}_{i=1}^{p}$ em que $q > p$ satisfaça essa propriedade.

A Figura 4.2 ilustra como três pontos em um plano podem ser divididos por um conjunto de funções lineares indicadoras, enquanto que quatro pontos não podem. Nesse caso a dimensão VC é igual ao número de parâmetros livres, mas em geral este não é o caso; i. e. a função $A\sin(bx)$ possui uma dimensão VC infinita (VAPNIK, 1995). De forma genérica, para funções lineares no $\mathbb{R}^n$, com $n \geq 2$, a dimensão VC é igual a $n + 1$.

Minimização de Risco Estrutural:

Cria-se uma estrutura tal que S_h é o espaço de hipótese de dimensão VC igual a h então (Equação (4.8)),

$$S_1 \subset S_2 \subset \cdots \subset S_\infty \tag{4.8}$$

O SRM consiste em resolver o seguinte problema (Equação (4.9))

$$\min_{S_h} R_{emp}[f] + \sqrt{\frac{h\ln\left(\frac{2l}{h}+1\right) - \ln\left(\frac{\delta}{4}\right)}{l}} \tag{4.9}$$

em que S_h é o espaço de hipótese de dimensão VC, h é a dimensão VC, l é a quantidade de pontos do conjunto a ser separado e δ é a probabilidade

Se o algoritmo do processo modelado é não-determinístico, ou seja, utiliza um valor externo que não o valor de entrada, o problema de modelagem torna-se menos preciso. Este capítulo está restrito a processos determinísticos, desta forma não serão abordados os processos não-determinísticos.

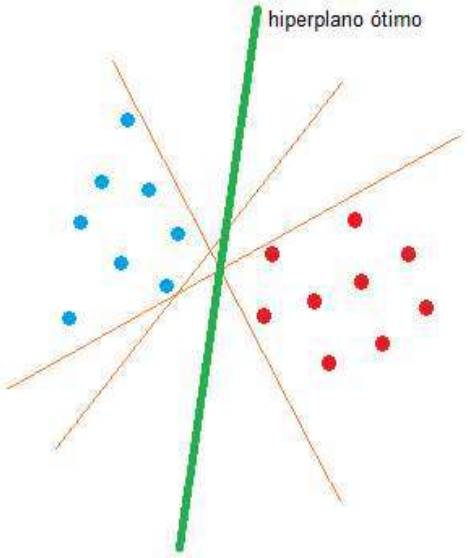
Os problemas de múltiplas saídas geralmente são capazes de serem reduzidos a um conjunto de problemas de única saída, sendo considerados independentes. Por isso, considera-se a existência de processos com múltiplas entradas e uma única saída.

4.3. Máquina de Vetores de Suporte

O problema de classificação pode ser restrito para consideração de problemas de duas classes sem perda de generalidade. Nesse problema o objetivo é separar duas classes por uma função que é inferida através de exemplos disponíveis. O objetivo é produzir um classificador que generaliza de forma satisfatória com amostras desconhecidas, separando-as de acordo com os vetores de suporte treinados pelo SVM. Considere o exemplo da Figura 4.3. Existem muitos possíveis classificadores lineares que podem separar os dados, mas só há um que maximiza a margem (maximiza a distância entre ele e os pontos de dados mais próximos de cada classe).

Este classificador linear é denominado o hiperplano ótimo de separação. Intuitivamente, espera-se que este limite separe os dados de maneira satisfatória em todas as situações, ao contrário de outros limites possíveis.

Figura 4.3 - Hiperplano ótimo de separação.



Considera-se o problema de separar os vetores de treinamento (**x**) pertencentes as duas classes a serem separadas (Equação (4.10)),

$$\mathcal{D} = \{(\mathbf{x}_1, y_1), \cdots, (\mathbf{x}_l, y_l)\}, \quad \mathbf{x} \in \mathbb{R}^n, y \in \{-1,1\} \tag{4.10}$$

com o hiperplano (Equação (4.11)),

$$\langle \mathbf{w}, \mathbf{x} \rangle + b = 0 \tag{4.11}$$

em que **x** são os vetores de dados a serem separados, **w** é a matriz de pesos, *b* o valor de *bias* e $\langle \mathbf{w}, \mathbf{x} \rangle$ é o produto interno entre **x** e **w**.

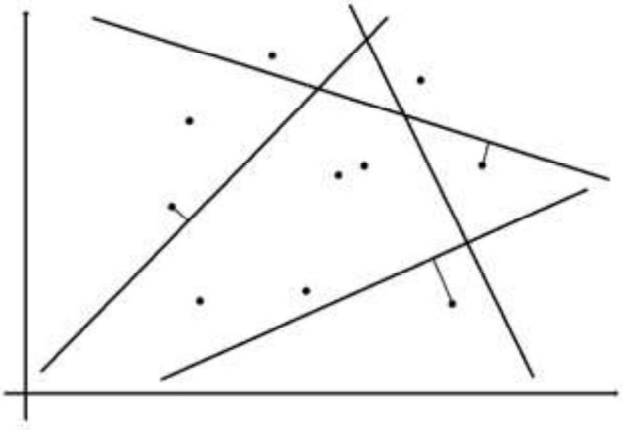
O conjunto de vetores é escolhido para ser otimamente separado pelo hiperplano se estiver afastado sem erro e com uma distância máxima entre o mais próximo vetor e o hiperplano. Existe alguma redundância na Equação (4.11, e sem perda de generalidade, é conveniente considerar um hiperplano canônico (VAPNIK, 1995), onde os parâmetros de **w**, *b* são limitados por (Equação (4.12)),

$$\min_i |\langle \mathbf{w}, \mathbf{x}_i \rangle + b| = 1 \tag{4.12}$$

em que $\mathbf{x}_i$ são os vetores de dados de *i*, para *i* = 1, 2, …, *l*.

Esta restrição incisiva na parametrização é preferível a outras alternativas no sentido de simplificar a formulação do problema. Em palavras declara-se que: *a norma do vetor peso deve ser igual ao inverso da distância do ponto mais próximo do conjunto de dados para o hiperplano*. A ideia é ilustrada na Figura 4.4, em que a distância do ponto mais próximo a cada hiperplano é mostrada.

Figura 4.4 - Hiperplano ótimo de separação - distância entre ponto e hiperplano.



Um hiperplano de separação na forma canônica deve satisfazer a seguinte restrição (Equação (4.13)),

$$y_i[\langle \mathbf{w}, \mathbf{x}_i \rangle + b] \geq 1, \quad i = 1, \cdots, l \tag{4.13}$$

A distância $d(\mathbf{w}, b; \mathbf{x})$ de um ponto $\mathbf{x}$ para o hiperplano $(\mathbf{w}, b)$ é (Equação (4.14)),

$$d(\mathbf{w}, b; \mathbf{x}) = \frac{|\langle \mathbf{w}, \mathbf{x}_i \rangle + b|}{\|\mathbf{w}\|} \tag{4.14}$$

O hiperplano ótimo é dado pela maximização da margem, ρ, sujeito as restrições da Equação (4.13). A margem é dada por (Equação (4.15)),

$$\begin{aligned} \rho(\mathbf{w}, b) &= \min_{\mathbf{x}_i : y_i = -1} d(\mathbf{w}, b; \mathbf{x}_i) + \min_{\mathbf{x}_i : y_i = -1} d(\mathbf{w}, b; \mathbf{x}_i) \\ &= \min_{\mathbf{x}_i : y_i = -1} \frac{|\langle \mathbf{w}, \mathbf{x}_i \rangle + b|}{\|\mathbf{w}\|} + \min_{\mathbf{x}_i : y_i = -1} \frac{|\langle \mathbf{w}, \mathbf{x}_i \rangle + b|}{\|\mathbf{w}\|} \\ &= \frac{1}{\|\mathbf{w}\|} \left(\min_{\mathbf{x}_i : y_i = -1} |\langle \mathbf{w}, \mathbf{x}_i \rangle + b| + \min_{\mathbf{x}_i : y_i = -1} |\langle \mathbf{w}, \mathbf{x}_i \rangle + b| \right) = \frac{2}{\|\mathbf{w}\|} \end{aligned} \tag{4.15}$$

Então o hiperplano que separa de forma ótima os dados é o que minimiza o Lagrangiano (Equação (4.16))

$$\Phi(\mathbf{w}) = \frac{1}{2}\|\mathbf{w}\|^2 \tag{4.16}$$

é independente de b porque na Equação (4.13) fornecida é satisfeita (ou seja, é um hiperplano de separação) mudando b, este se moverá na direção normal para si. Por conseguinte, a margem do se mantém inalterada, mas o hiperplano já não é o ótimo na medida em que ficará mais perto de uma classe do que de outra. Minimizar a Equação (4.16) é equivalente à aplicação do princípio SRM. Suponha que o limite seguinte detém (Equação (4.17)),

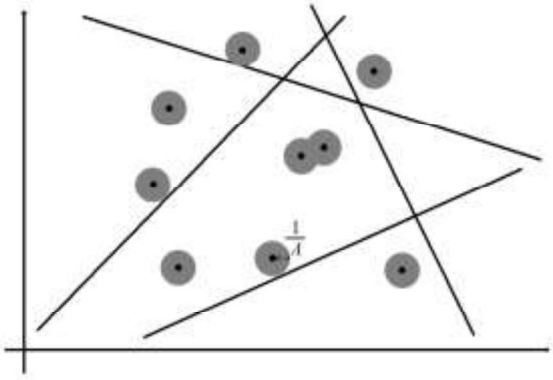
$$\|\mathbf{w}\| < A \tag{4.17}$$

em que A é um limite arbitrário para a norma de **w**. Então das Equações (4.13) e (4.14), se tornam (Equação (4.18)),

$$d(\mathbf{w}, b; \mathbf{x}) \geq \frac{1}{A} \tag{4.18}$$

Em conformidade, os hiperplanos não podem estar mais perto do que $1/A$ a qualquer um dos pontos de dados. Intuitivamente, pode ser visto na Figura 4.5 como a distância $1/A$ reduz os possíveis hiperplanos e, portanto, a capacidade.

Figura 4.5 - Restrição dos hiperplanos canônicos.



A dimensão VC, h, do conjunto de hiperplanos canônicos no espaço dimensional n é delimitada por (Equação (4.19)),

$$h \leq \min\left[R_{he}{}^2 A^2, n\right] + 1 \tag{4.19}$$

em que R_{he} é o raio de uma hiperesfera abrangendo todos os pontos. Minimizando assim a Equação (4.16) é equivalente a minimizar um limite superior sobre a dimensão VC. A solução

para o problema de otimização da Equação (4.16) sob as restrições da Equação (4.13) é dada pelo ponto de sela do funcional de Lagrange (Lagrangiano) (MINOUX, 1986) (Equação (4.20)).

$$\Phi(\mathbf{w}, b, \alpha) = \frac{1}{2}\|\mathbf{w}\|^2 - \sum_{i=1}^{l} \alpha_i (y_i[\langle \mathbf{w}, \mathbf{x}_i \rangle + b] - 1) \tag{4.20}$$

em que α são os multiplicadores de Lagrange. O Lagrangiano tem que ser minimizado a respeito de $\mathbf{w}$ e b e maximizado com respeito a $\alpha \geq 0$. A dualidade do Lagrangiano Clássico permite que o problema primordial, a Equação (4.20), seja transformado para o seu problema duplo, que tem mais fácil resolução. O problema duplo é dado por (Equação (4.21)),

$$\max_{\alpha} \mathbf{W}(\alpha) = \max_{\alpha} \left(\min_{\mathbf{w},b} \Phi(\mathbf{w}, b, \alpha) \right) \tag{4.21}$$

O mínimo a respeito do $\mathbf{w}$ e b do Lagrangiano, Φ, é dado por (Equação (4.22)),

$$\begin{aligned} \frac{\partial \Phi}{\partial b} = 0 \Rightarrow \sum_{i=1}^{l} \alpha_i y_i = 0 \\ \frac{\partial \Phi}{\partial \mathbf{w}} = 0 \Rightarrow \mathbf{w} = \sum_{i=1}^{l} \alpha_i y_i \mathbf{x}_i \end{aligned} \tag{4.22}$$

Daí a partir das Equações (4.20) a (4.22), o problema duplo é (Equação (4.23)),

$$\max_{\alpha} \mathbf{W}(\alpha) = \max_{\alpha} -\frac{1}{2} \sum_{i=1}^{l} \sum_{j=1}^{l} \alpha_i \alpha_j y_i y_j \langle \mathbf{x}_i, \mathbf{x}_j \rangle + \sum_{k=1}^{l} \alpha_k, \tag{4.23}$$

e, portanto, a solução para o problema é dada por (Equação (4.24)),

$$\alpha^* = \arg\min_{\alpha} \frac{1}{2} \sum_{i=1}^{l} \sum_{j=1}^{l} \alpha_i \alpha_j y_i y_j \langle \mathbf{x}_i, \mathbf{x}_j \rangle - \sum_{k=1}^{l} \alpha_k, \tag{4.24}$$

com restrições (Equação (4.25)),

$$\begin{aligned} \alpha_i &\geq 0, \quad i = 1, \cdots, l \\ \sum_{j=1}^{l} \alpha_j y_j &= 0 \,. \end{aligned} \tag{4.25}$$

Resolvendo a Equação (4.24) com as restrições da Equação (4.25) determinam-se os multiplicadores de Lagrange e o hiperplano ótimo de separação é dado por (Equação (4.26)),

$$\begin{aligned} \mathbf{w}^* &= \sum_{i=1}^{l} \alpha_i y_i \mathbf{x}_i \\ b^* &= -\frac{1}{2} \langle \mathbf{w}^*, \mathbf{x}_r + \mathbf{x}_s \rangle \end{aligned} \tag{4.26}$$

em que x_r e x_s são quaisquer vetores de suporte para cada classe que satisfaz (Equação (4.27)),

$$\alpha_r, \alpha_s > 0, \quad y_r = -1, y_s = 1 \tag{4.27}$$

O classificador rígido é então (Equação (4.28)),

$$f(\mathbf{x}) = sgn(\langle \mathbf{w}^*, \mathbf{x} \rangle + b) \tag{4.28}$$

Alternativamente, um classificador atenuado pode ser utilizado para que interpole linearmente a margem (Equação (4.29)),

$$f(\mathbf{x}) = h(\langle \mathbf{w}^*, \mathbf{x} \rangle + b) \quad em\ que\ \ h(z) = \begin{cases} -1 & : \quad z < -1 \\ z & : \quad -1 \leq z \leq 1 \\ +1 & : \quad z > 1 \end{cases} \tag{4.29}$$

Esta equação pode ser mais apropriada do que o classificador rígido da Equação (4.28), porque produz uma saída de valores reais entre -1 e 1 quando o classificador é consultado dentro da margem em que não existirem dados de treinamento residentes. A partir das condições de Kuhn-Tucker (Equação (4.30)),

$$\alpha_i (y_i [\langle \mathbf{w}, \mathbf{x}_i \rangle + b] - 1) = 0, \quad i = 1, \cdots, l \tag{4.30}$$

e, portanto, apenas os pontos $\mathbf{x}_i$ que satisfazem a Equação (4.31),

$$y_i [\langle \mathbf{w}, \mathbf{x}_i \rangle + b] = 1 \tag{4.31}$$

que terá multiplicadores de Lagrange não nulos. Esses pontos são chamados de vetores de suporte (*Support Vectors* - SV). Se os dados são linearmente separáveis, todos os SV irão situar-se na margem e, portanto, a quantidade de SV pode ser muito pequena. Por conseguinte, o hiperplano é determinado por um pequeno subconjunto do conjunto de treinamento. Os outros pontos podem ser removidos do conjunto de treinamento e, recalculando o hiperplano, a resposta seria a mesma. Assim, a SVM pode ser usada para resumir a informação contida num conjunto de dados produzido pelos SV. Se os dados forem linearmente separáveis, a seguinte igualdade se manterá (Equação (4.32)),

$$\|\mathbf{w}\|^2 = \sum_{i=1}^{l} \alpha_i = \sum_{i \in SVs} \alpha_i = \sum_{i \in SVs} \sum_{j \in SVs} \alpha_i \alpha_j y_i y_j \langle \mathbf{x}_i, \mathbf{x}_j \rangle \tag{4.32}$$

Assim, a partir da Equação (4.27) a dimensão VC do classificador é delimitada por (Equação (4.33)),

$$h \leq \min \left[R_{he}^2 \sum_{i \in SVs} \alpha_i^*, n \right] + 1 \tag{4.33}$$

e, se os dados de treinamento x forem normalizados para residir no intervalo $[-1,+1]^n$, define-se um hipercubo unitário (Equação (4.34)),

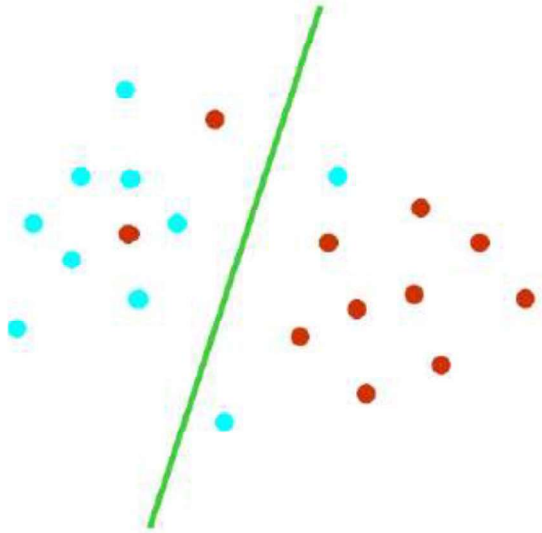
$$h \leq 1 + n \times \min \left[\sum_{i \in SVs} \alpha_i^*, 1 \right] \tag{4.34}$$

4.3.1. O Hiperplano Ótimo Generalizado de Separação

Até o presente momento a discussão ficou restrita aos casos em que os dados de treinamento são linearmente separáveis. Entretanto, de modo geral, esse não será o caso, como apresenta a Figura 4.6. Existem duas aproximações para a generalização do problema que dependem do conhecimento prévio do mesmo e da estimativa do ruído dos dados.

No caso em que se espera (e/ou se saiba) que um hiperplano possa corretamente separar os dados, o método de introduzir uma função custo adicional associada aos erros de classificação, é apropriado. Como alternativa, uma função mais complexa pode ser utilizada para descrever a margem, como fora discutido na Seção 4.3.

Figura 4.6 - Hiperplano ótimo generalizado de separação.



Para possibilitar que o método do hiperplano ótimo de separação seja generalizado, Cortes e Vapnik (1995) introduziram variáveis não negativas, $\xi_i \geq 0$, e uma função de penalidade (Equação (4.35)),

$$F_\sigma(\xi) = \sum_i \xi_i^\sigma, \quad \sigma > 0, \tag{4.35}$$

em que ξ_i é a medida dos erros de classificação. O problema de otimização é posto de forma a minimizar os erros de classificação, assim como minimizar a margem na dimensão VC do classificador. As restrições da Equação (4.13) são modificadas nos casos não separáveis para a Equação (4.36),

$$y_i[\langle \mathbf{w}, \mathbf{x}_i \rangle + b] \geq 1 - \xi_i, \quad i = 1, \cdots, l \tag{4.36}$$

em que $\xi_i \geq 0$. O hiperplano ótimo de separação generalizado é determinado pelo vetor $\mathbf{w}$, que minimiza o funcional (Equação (4.37)),

$$\Phi(\mathbf{w}, \xi) = \frac{1}{2}\|\mathbf{w}\|^2 + C\sum_i \xi_i \tag{4.37}$$

em que C é o valor dado sujeito às restrições da Equação (4.36). A solução para o problema de otimização da Equação (4.37) sob as restrições da Equação (4.36) é dada pelo ponto de sela do Lagrangiano (MINOUX, 1986) (Equação (4.38)),

$$\Phi(\mathbf{w}, b, \alpha, \xi, \beta) = \frac{1}{2}\|\mathbf{w}\|^2 + C\sum_i \xi_i - \sum_{i=1}^{l} \alpha_i (y_i[\mathbf{w}^T\mathbf{x}_i + b] - 1 + \xi_i) - \sum_{j=1}^{l} \beta_i \xi_i \qquad (4.38)$$

em que α, β são os multiplicadores de Lagrange. O Lagrangiano deve ser minimizado a respeito de $\mathbf{w}$, b, ξ e maximizado a respeito de α, β. Como antes, a dualidade Lagrangiana clássica permite que o problema primordial, apresentado na Equação (4.38), seja transformado no problema duplo. O problema duplo é dado pela Equação (4.39),

$$\max_{\alpha} \mathbf{W}(\alpha, \beta) = \max_{\alpha,\beta} \left(\min_{\mathbf{w},b,\xi} \Phi(\mathbf{w}, b, \alpha, \xi, \beta) \right) \qquad (4.39)$$

O mínimo a respeito de $\mathbf{w}$, b e ξ do Lagrangiano, Φ, é dado pela Equação (4.40),

$$\begin{aligned} \frac{\partial \Phi}{\partial b} = 0 \quad &\Rightarrow \quad \sum_{i=1}^{l} \alpha_i y_i = 0 \\ \frac{\partial \Phi}{\partial \mathbf{w}} = 0 \quad &\Rightarrow \mathbf{w} = \sum_{i=1}^{l} \alpha_i y_i \mathbf{x}_i \\ \frac{\partial \Phi}{\partial \xi} = 0 \quad &\Rightarrow \quad \alpha_i + \beta_i = C \end{aligned} \qquad (4.40)$$

Portanto, a partir das Equações (4.38) a (4.40), o problema é duplo, como apresentado na Equação (4.41),

$$\max_{\alpha} \mathbf{W}(\alpha) = \max_{\alpha} -\frac{1}{2}\sum_{i=1}^{l}\sum_{j=1}^{l} \alpha_i \alpha_j y_i y_j \langle \mathbf{x}_i, \mathbf{x}_j \rangle + \sum_{k=1}^{l} \alpha_k \qquad (4.41)$$

sendo assim, a solução do problema é dada por (Equação (4.42)),

$$\alpha^* = \arg\min_{\alpha} \frac{1}{2}\sum_{i=1}^{l}\sum_{j=1}^{l} \alpha_i \alpha_j y_i y_j \langle \mathbf{x}_i, \mathbf{x}_j \rangle - \sum_{k=1}^{l} \alpha_k \qquad (4.42)$$

com as restrições (Equação (4.43)),

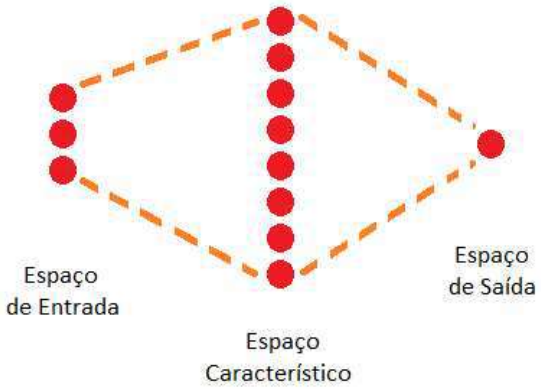
$$\begin{aligned} 0 \le \alpha_i \le C \quad i = 1, \cdots, l \\ \sum_{j=1}^{l} \alpha_j y_j = 0 \end{aligned} \qquad (4.43)$$

A solução do problema de minimização e do caso separável se mostra idêntica, exceto pela modificação das margens dos multiplicadores de Lagrange. A incerteza existente dentro da aproximação de Cortes é que o coeficiente C deve ser determinado. Este parâmetro introduz uma capacidade de controle adicional ao classificador. O coeficiente C pode ser diretamente relacionado com o parâmetro de regularização (GIROSI, 1997; SMOLA; SCHÖLKOPF, 1997). Blanz e colaboradores (1996) utilizam o valor de $C = 5$, mas finalmente C deve ser escolhido para refletir o conhecimento do ruído presente nos dados.

4.4. Generalização em Espaço Característico de Alta Dimensão

No caso em que a margem linear é inapropriada, o SVM pode mapear o vetor de entrada, **x**, dentro de um espaço característico de alta dimensão, z. Ao escolher um mapeamento não linear, a priori, o SVM calcula um hiperplano ótimo separando neste espaço de maior dimensão, como esquematizado na Figura 4.7. A ideia explora o método de Aizerman e colaboradores (1964) a qual habilita a dimensionalidade (BELLMAN, 1961) a ser utilizada.

Figura 4.7 - Mapeamento do espaço de entrada em um espaço característico de alta dimensão.



Existem algumas restrições que podem ser aplicadas ao mapeamento não linear, como pode-se observar na Seção 4.3.1, entrentato, as funções mais comumente empregadas já são aceitáveis ao mapeamento não linear. Entre os mapeamentos aceitáveis existem os polinomiais, as funções de base radial e certas funções sigmóide. O problema de otimização da Equação (4.42) torna-se (Equação (4.44)).

$$\alpha^* = \arg\min_{\alpha} \frac{1}{2}\sum_{i=1}^{l}\sum_{j=1}^{l}\alpha_i\alpha_j y_i y_j K(\mathbf{x},\mathbf{x}') - \sum_{k=1}^{l}\alpha_k \tag{4.44}$$

em que $K(\mathbf{x},\mathbf{x}')$ é a função núcleo (*kernel*) realizando o mapeamento não linear no espaço característico com restrições não modificadas de acordo com a Equação (4.45),

$$0 \leq \alpha_i \leq C \quad i = 1, \cdots, l$$
$$\sum_{j=1}^{l} \alpha_j y_j = 0 \tag{4.45}$$

Resolvendo a Equação (4.44) com as restrições da Equação (4.45) encontram-se os multiplicadores de Lagrange. O classificador SVM rígido que implementa um hiperplano ótimo de separação no espaço característico é ilustrado pela Equação (4.46),

$$f(x) = sgn\left(\sum_{i \in SVs} \alpha_i y_i K(\mathbf{x}_i, \mathbf{x}) + b^*\right) \tag{4.46}$$

em que (Equação (4.47))

$$\langle \mathbf{w}^*, \mathbf{x} \rangle = \sum_{i=1}^{l} \alpha_i y_i K(\mathbf{x}_i, \mathbf{x})$$
$$b^* = -\frac{1}{2} \sum_{i=1}^{l} \alpha_i y_i [K(\mathbf{x}_i, \mathbf{x}_r) + K(\mathbf{x}_i, \mathbf{x}_s)] \tag{4.47}$$

A tendência é calculada aqui utilizando dois vetores de suporte, mas pode ser calculada utilizando todos os SV sobre a margem para estabilidade (VAPNIK *et al*., 1997). Se o núcleo possui um termo de tendência (*bias*), ele pode ser acomodado dentro do núcleo, desta forma o classificador é simplificado para a Equação (4.48),

$$f(x) = sgn\left(\sum_{i \in SVs} \alpha_i K(\mathbf{x}_i, \mathbf{x})\right) \tag{4.48}$$

Muitos núcleos aplicados possuem o termo de tendência e qualquer núcleo finito pode ser modificado para ter um também um termo de tendêndia (GIROSI, 1997). Isso simplifica o problema de otimização através da remoção das restrições de igualdade da Equação (4.45). A Seção 4.5 do presente trabalho traz a discussão acerca das condições necessárias para que a validade das funções núcleo sejam efetivadas.

Comentários

Tipicamente, os dados só serão linearmente separáveis em alguns espaços característicos possivelmente com uma dimensão muito elevada. Pode não ser possível separar os dados de maneira adequada, especialmente quando apenas uma quantidade finita de dados de treinamento está disponível, caso esses dados estejam potencialmente corrompidos por ruído. Por isso, na prática, será necessário empregar uma abordagem não separável que coloca um limite superior nos multiplicadores de Lagrange. Isso levanta a questão de como se deve determinar o parâmetro C. Este parâmetro é definido como o custo para a classificação dos dados pelo SVM, quanto menor seu valor, menor a possibilidade da existência de *outliers*, que são dados que possam ser classificados de maneira errônea. O processo se assemelha ao problema de regularização em que o coeficiente de regularização tem de ser determinado.

O parâmetro C pode ser diretamente relacionado ao parâmetro de regularização para certos núcleos como demostraram Smola e Schölkopf (1997). Um processo de validação cruzada pode ser utilizado para determinar este parâmetro, embora métodos mais eficientes e potencialmente melhores ainda não foram definidos e constituem objeto de pesquisas. Na remoção de padrões dos dados de treinamento que não são vetores de suporte, a solução mantém-se inalterada e, portanto, um método rápido para validação pode estar disponível quando os vetores de suporte são escassos.

4.5. Espaço Característico

Nesta seção se discute o método utilizado para mapear os dados de entrada em um espaço característico de alta dimensão através do uso de núcleos. A ideia da função núcleo é permitir que operações sejam executadas no espaço de entrada ao invés do espaço característico de alta dimensão. Por isso, o produto interno não necessita ser avaliado no espaço característico. Isso fornece uma maneira de lidar com a dimensionalidade, no entanto, o cálculo é ainda dependente do número de padrões de formação do mapeamento. Afim de proporcionar uma maior qualidade à distribuição de dados para um problema de grande dimensão, faz-se necessário a utilização de um grande conjunto de dados de treinamento.

Funções núcleo

A teoria de funções núcleo é baseada nos Espaços Núcleo de Hilbert (*Reproducing Kernel Hilbert Spaces* – RKHS) (ARONSZAJN, 1950; GIROSI, 1997; HECKMAN, 1997; WAHBA, 1990). O produto interno no espaço característico possui um núcleo equivalente no espaço de entrada da Equação (4.49),

$$K(\mathbf{x}, \mathbf{x}') = \langle \phi(\mathbf{x}), \phi(\mathbf{x}') \rangle \tag{4.49}$$

desde que certas condições sejam mantidas. Se Ké uma função positiva simétrica definida, o que satisfaz as Condições de Mercer, apresentadas nas Equações (4.50) e (4.51),

$$K(\mathbf{x}, \mathbf{x}') = \sum_{m}^{\infty} a_m \phi_m(\mathbf{x}) \phi_m(\mathbf{x}'), \quad a_m \geq 0 \tag{4.50}$$

$$\iint K(\mathbf{x}, \mathbf{x}') g(\mathbf{x}) g(\mathbf{x}') d\mathbf{x} d\mathbf{x}' > 0, \quad g \in L_2 \tag{4.51}$$

em que g é uma função definida pelo tipo de função núcleo. Desta forma o núcleo representa um legítimo produto interno no espaço característico. Funções válidas que satisfaçam as condições de Mercer serão apresentadas nos próximos tópicos. As funções núcleo seguintes são válidas para todo $\mathbf{x}$ e $\mathbf{x}'$ reais, com ressalvas a existência de algumas exceções.

Função Núcleo Polinomial

O mapeamento polinomial é um método popular para modelagem não linear, como apresentado nas Equações (4.52) e (4.53),

$$K(\mathbf{x}, \mathbf{x}') = \langle \mathbf{x}, \mathbf{x}' \rangle^d \tag{4.52}$$

$$K(\mathbf{x}, \mathbf{x}') = (\langle \mathbf{x}, \mathbf{x}' \rangle + 1)^d \tag{4.53}$$

Função Núcleo de Base Radial Gaussiana

Funções de base radial têm recebido atenção significativa, mais comumente na forma Gaussiana (Equação (4.54)),

$$K(\mathbf{x}, \mathbf{x}') = \exp\left(-\frac{\|\mathbf{x} - \mathbf{x}'\|^2}{2\sigma^2} \right) \tag{4.54}$$

em que σ é o desvio padrão dos dados a serem mapeados.

As clássicas técnicas utilizando funções de base radial empregam algum método para se determinar um subconjunto de núcleos. Frequentemente, um método de agrupamento é utilizado primeiramente para selecionar um subconjunto de núcleos.

Uma característica atrativa do SVM está na seleção implícita de subconjuntos de núcleos em que cada vetor de suporte coincide com uma função Gaussiana local centrada naquele ponto dos dados (vetor de suporte). É possível selecionar a largura da função de base global, s, utilizando o princípio de SRM (VAPNIK, 1995).

Função Núcleo de Base Radial Exponencial

A função de base radial da forma em que é apresentada na Equação (4.55),

$$K(\mathbf{x},\mathbf{x}') = \exp\left(-\frac{\|\mathbf{x}-\mathbf{x}'\|}{2\sigma^2}\right) \tag{4.55}$$

produz uma solução linear por partes, que pode ser atrativa quando descontinuidades são aceitáveis.

Função Núcleo por Séries de Fourier

Uma série de Fourier pode ser considerada uma expansão do espaço característico seguinte de dimensão $2N+1$. O núcleo é definido no intervalo $\left[-\frac{\pi}{2},\frac{\pi}{2}\right]$ (Equação (4.56)),

$$K(\mathbf{x},\mathbf{x}') = \frac{sen\left(N+\frac{1}{2}\right)(\mathbf{x}-\mathbf{x}')}{sen\left(\frac{1}{2}(\mathbf{x}-\mathbf{x}')\right)} \tag{4.56}$$

Entretanto, este núcleo possa não ser uma boa opção visto que sua capacidade de regularização é ruim, devido à sua transformada de Fourier (SMOLA; SCHÖLKOPF, 1997).

Função Núcleo *Splines*

Splines é uma escolha popular para modelagem devido a sua flexibilidade (DIERCKX, 1993). Um *spline* finito, de ordem κ, com N nós localizados em τ_s é apresentado de acordo com a Equação (4.57),

$$K(\mathbf{x},\mathbf{x}') = \sum_{r=0}^{\kappa}\mathbf{x}^r\mathbf{x}'^r + \sum_{s=1}^{N}(\mathbf{x}-\tau_s)_+^{\kappa}(\mathbf{x}'-\tau_s)_+^{\kappa} \tag{4.57}$$

Um *spline* infinito é definido no intervalo $[0,1)$ dado por (Equação (4.58)),

$$K(\mathbf{x},\mathbf{x}') = \sum_{r=0}^{\kappa}\mathbf{x}^r\mathbf{x}'^r + \int_0^1(\mathbf{x}-\tau_s)_+^{\kappa}(\mathbf{x}'-\tau_s)_+^{\kappa}d\tau \tag{4.58}$$

No caso em que $\kappa = 1$, (S_1^{∞}), o núcleo é dado por (Equação (4.59)),

$$K(\mathbf{x},\mathbf{x}') = 1 + \langle \mathbf{x},\mathbf{x}' \rangle + \frac{1}{2}\langle \mathbf{x},\mathbf{x}' \rangle \min(\mathbf{x},\mathbf{x}') - \frac{1}{6}\min(\mathbf{x},\mathbf{x}')^3 \tag{4.59}$$

em que a solução é cúbica por partes.

Função Núcleo *Splines* B

A função núcleo *Splines* B é outra formulação popular da função núcleo *spline* (DIERCKX, 1993). O núcleo é definido no intervalo $[-1,1]$, e possui uma forma fechada apresentada na Equação (4.60),

$$K(\mathbf{x},\mathbf{x}') = B_{2N+1}(\mathbf{x}-\mathbf{x}') \tag{4.60}$$

Existem ainda algumas maneiras de se utilizar as funções núcleo, utilizando alguns recursos, tais como:

Núcleo Aditivo

Núcleos mais complexos podem ser obtidos através da formação de núcleos de soma, uma vez que a soma de duas funções positiva definidas também é positiva definida (Equação (4.61)).

$$K(\mathbf{x},\mathbf{x}') = \sum_i K_i(\mathbf{x},\mathbf{x}') \tag{4.61}$$

Produto Tensor

Núcleos multidimensionais podem ser obtidos através da formação de produtos tensores de núcleos (ARONSZAJN, 1950) (Equação (4.62)),

$$K(\mathbf{x},\mathbf{x}') = \prod_i K_i(\mathbf{x},\mathbf{x}') \tag{4.62}$$

A utilização do produto tensor é particularmente útil na construção de núcleos de *spline* multidimensionais, que são obtidos a partir do produto de núcleos univariados.

Normalização dos dados

A normalização de dados é necessária para alguns núcleos em particular devido ao seu domínio restrito e pode ser vantajosa também para o núcleo sem restrições. Para determinar se a normalização (isotrópica ou não isotrópica) de dados é necessária, leva-se em consideração

as características da entrada. Além disso, a normalização vai tornar a matriz Hessiânica mais uniforme facilitando a solução do problema de otimização.

Seleção de núcleo

Para a seleção de núcleo surge a seguinte questão: com tantas opções de mapeamentos distintos disponíveis para serem escolhidos, qual seria dentre esses o melhor para um determinado problema? Esta questão não é nova, entretanto, com a inclusão de muitos mapeamentos em um único âmbito fica mais fácil realizar uma comparação. O limite superior da dimensão VC, apresentado na Equação (4.19), revela-se um caminho em potencial para fornecer um meio de compação dos núcleos. No entanto, para a obtenção do limite superior da dimensão VC, é necessário calcular uma estimativa do raio da hiperesfera que reúne os dados no espaço característico não linear. Por fim, como precaução, mesmo que seja desenvolvido um método teórico ideal para seleção do núcleo, a menos que este método possa ser validado usando conjuntos de treinamento independentes para um grande número de problemas, tais como os métodos de inicialização e de validação cruzada, permanecerão como métodos preferidos para a seleção do núcleo os de inicialização e de validação cruzada.

4.6.Máquinas de Vetores de Suporte de Classificação

A seguinte formulação da minimização abaixo (Equação (4.63)) apresenta a função objetivo do classificador SVM.

$$\min_{\mathbf{w},b,\xi_i} \frac{\|\mathbf{w}\|^2}{2} + C\sum_{i=1}^{n}\xi_i \tag{4.63}$$

sujeito à (Equação (4.64))

$$\begin{aligned} \mathrm{y}_i(\mathbf{w}^T\phi(\mathbf{x}_i) + b) &\geq 1 - \xi_i \quad \text{para todo} \quad i = 1, \ldots, n \\ \xi_i &\geq 0 \quad \text{para todo} \quad i = 1, \ldots, n \end{aligned} \tag{4.64}$$

Quando este problema de minimização (com programação quadrática) é resolvido utilizando multiplicadores de Lagrange, a função de decisão (classificação) para os dados *x* se torna a Equação (4.65).

$$f(x) = \operatorname{sgn}\left(\sum_{i=1}^{n}\alpha_i y_i K(\mathbf{x}, \mathbf{x}_i) + b\right) \tag{4.65}$$

em que α_i é o multiplicador de Lagrange. Todo ponto com $\alpha_i > 0$ é ponderado na função de decisão e fornece suporte à máquina de vetores. Existem relativamente poucos multiplicadores de Lagrange com valor não-nulo, tornando o número de vetores de suporte não excessivo. Para o método de classificação *one class*, o rótulo dos dados (y_i) assume valor +1 para dados que façam parte da classe, enquanto para os dados que não façam parte do grupo, o valor do rótulo é –1.

A mais antiga implementação para classificação multiclasses através de SVM é provavelmente o método *one-against-all* (um contra todos, OAA). Todo classificador multiclasses deve ser capaz de separar os dados entre *k* classes, sendo *k* > 2. São construídos *k* modelos SVM em que *k* é o número de classes. O modelo SVM *m*-ésimo é treinado com todos os dados da *m*-ésima classe com rótulos positivos (+1), e todos os outros dados com rótulos negativos (-1). Desta forma, fornecidos os *l* dados de treinamento $(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_l, y_l)$, em que $\mathbf{x}_i \in \mathbb{R}^n, i = 1, \ldots, l$ e $y_i \in \{1, \ldots, k\}$ é a classe de $\mathbf{x}_i$, o *m*-ésimo modelo SVM resolve o seguinte problema (Equação (4.66)).

$$\begin{aligned} \min_{\mathbf{w}^m, b^m, \xi^m} \quad & \frac{1}{2}(\mathbf{w}^m)^T\mathbf{w}^m + C\sum_{i=1}^{l}\xi_i^m \\ & (\mathbf{w}^m)^T\phi(\mathbf{x}_i) + b^m \geq 1 - \xi_i^m, \quad \text{se } y_i = m \\ & (\mathbf{w}^m)^T\phi(\mathbf{x}_i) + b^m \leq -1 - \xi_i^m, \quad \text{se } y_i \neq m \\ & \xi_i^m \geq 0, \quad i = 1, \ldots, l, \end{aligned} \tag{4.66}$$

em que o dado de treinamento $\mathbf{x}_i$ é mapeado para um espaço de maior dimensão pela função ϕ e C é o parâmetro de penalidade do SVM. Minimizar $\frac{1}{2}(\mathbf{w}^m)^T\mathbf{w}^m$ significa que se deseja maximizar $2/\|\mathbf{w}^m\|$, que constitui uma margem entre dois grupos de dados. Quando os dados não são linearmente separáveis, o termo de penalidade $C\sum_{i=1}^{l}\xi_i^m$ pode reduzir o número de erros de treinamento. O conceito básico por trás do SVM é encontrar um equilíbrio entre o termo de regularização $\frac{1}{2}(\mathbf{w}^m)^T\mathbf{w}^m$ e o termo de penalidade.

Após a solução da Equação (4.66), obtém-se *k* funções de decisão (Equação (4.67)):

$$\begin{aligned} & (\mathbf{w}^1)^T\phi(\mathbf{x}) + b^1 \\ & \vdots \\ & (\mathbf{w}^k)^T\phi(\mathbf{x}) + b^k. \end{aligned} \tag{4.67}$$

Diz-se que *x* está na classe que possui o maior valor para a função de decisão (Equação (4.68)):

$$\text{classe de } \mathbf{x} \equiv \underset{m=1,\ldots,\mathrm{k}}{\operatorname{argmax}}((\mathbf{w}^m)^T \phi(\mathbf{x}) + b^m) \tag{4.68}$$

São resolvidos o problema quadrático dado pela Equação (4.66) *k* vezes com *l* variáveis.

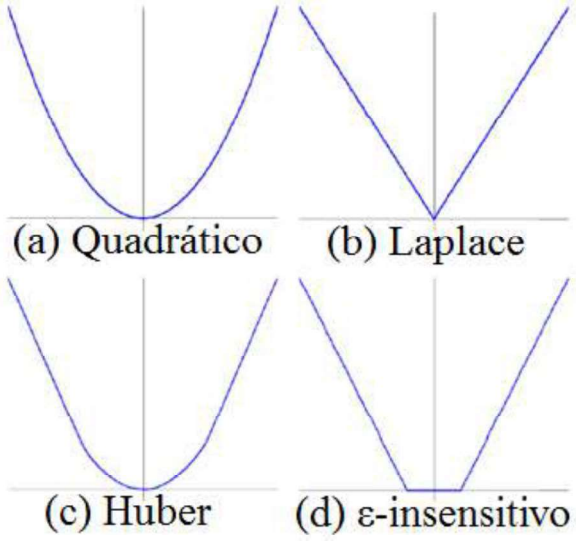
Outro método importante de classificação indireta é o chamado *one-against-one* (um contra um, OAO) que fora introduzido por Knerr e colaboradores (1990), e as primeiras utilizações dessa estratégia no SVM foram realizadas por Friedman (1996) e Kressel (1998). Neste método deve-se construir $k(k-1)/2$ classificadores em que cada um deles é treinado com dados de duas classes distintas. Para os dados de treinamento da *i*-ésima e *j*-ésima classes, deve-se resolver o seguinte problema de classificação binário (Equação (4.69)):

$$\begin{aligned} \min_{\mathbf{w}^{ij}, b^{ij}, \xi^{ij}} \quad & \frac{1}{2}\left(\mathbf{w}^{ij}\right)^T \mathbf{w}^{ij} + C \sum_t \xi_t^{ij} \\ & \left(\mathbf{w}^{ij}\right)^T \phi(\mathbf{x}_t) + b^{ij} \geq 1 - \xi_i^{ij}, \quad \text{se} \quad y_t = i \\ & \left(\mathbf{w}^{ij}\right)^T \phi(\mathbf{x}_t) + b^{ij} \leq -1 - \xi_i^{ij}, \quad \text{se} \quad y_t = j \\ & \xi_t^{ij} \geq 0. \end{aligned} \tag{4.69}$$

4.7. Máquina de Vetores de Suporte para Regressão

Os SVMs podem também ser aplicados para problemas de regressão através da introdução de uma função de perda (*L*) alternativa (SMOLA, 1996). A função *L* deve ser modificada de forma a incluir a medida de distância. A Figura 4.8 ilustra quatro possíveis funções de perda.

Figura 4.8 - Funções de perda.



A função de perda da Figura 4.8 (a) corresponde ao critério convencional de erro dos

mínimos quadrados. A função da Figura 4.8 (b) é a função de perda Laplaciana que é menos sensível a valores discrepantes que a função de perda quadrática. Huber propôs a função de perda na Figura 4.8 (c), como uma função de perda robusta que tem propriedades ótimas quando a distribuição subjacente dos dados é desconhecida. Estas três funções de perda não produzirão insuficiência de vetores de suporte. Para abordar esta questão, Vapnik propôs a função de perda da Figura 4.8 (d) como uma aproximação da função de perda de Huber que permite que um conjunto esparso de vetores de suporte seja obtido.

Regressão Linear

Considere o problema de aproximação do conjunto de dados (Equação (4.70)),

$$\mathcal{D} = \{(\mathbf{x}_1, y_1), \cdots, (\mathbf{x}_l, y_l)\}, \quad \mathbf{x} \in \mathbb{R}^n, y \in \mathbb{R}, \tag{4.70}$$

com a função linear (Equação (4.71)),

$$f(\mathbf{x}) = \langle \mathbf{w}, \mathbf{x} \rangle + b, \tag{4.71}$$

a função de regressão ótima é dada pelo mínimo do funcional (Equação (4.72)),

$$\Phi(\mathbf{w}, \xi) = \frac{1}{2}\|\mathbf{w}\|^2 + C\sum_i (\xi_i^- + \xi_i^+), \tag{4.72}$$

em que C é um valor pré-especificado, e ξ_i^- e ξ_i^+ são variáveis folga representando restrições inferiores e superiores, respectivamente, nas saídas do sistema.

Função de Perda ε-insensitiva

Utilizando uma função de perda ε-insensitiva, como da Figura 4.8 (d) (Equação (4.73)),

$$L_\varepsilon(y) = \begin{cases} 0 & \text{para} \quad |f(\mathbf{x}) - y| - \varepsilon \\ |f(\mathbf{x}) - y| - \varepsilon & \text{de outro modo} \end{cases} \tag{4.73}$$

a solução é dada por (Equação (4.74)),

$$\max_{\alpha,\alpha^*} \mathbf{W}(\alpha,\alpha^*) = \max_{\alpha,\alpha^*} -\frac{1}{2}\sum_{i=1}^{l}\sum_{j=1}^{l}(\alpha_i-\alpha_i^*)\left(\alpha_j-\alpha_j^*\right)\langle \mathbf{x}_i,\mathbf{x}_j\rangle + \sum_{i=1}^{l}\alpha_i(y_i-\varepsilon)-\alpha_i^*(y_i+\varepsilon) \tag{4.74}$$

ou alternativamente (Equação (4.75)),

$$\bar{\alpha},\bar{\alpha}^* = \arg\min_{\alpha,\alpha^*}\frac{1}{2}\sum_{i=1}^{l}\sum_{j=1}^{l}(\alpha_i-\alpha_i^*)\left(\alpha_j-\alpha_j^*\right)\langle \mathbf{x}_i,\mathbf{x}_j\rangle - \sum_{i=1}^{l}(\alpha_i-\alpha_i^*)y_i + \sum_{i=1}^{l}(\alpha_i+\alpha_i^*)\varepsilon \tag{4.75}$$

com as restrições (Equação (4.76)),

$$0 \le \alpha_i,\alpha_i^* \le C, \quad i=1,\cdots,l \qquad \sum_{i=1}^{l}(\alpha_i-\alpha_i^*)=0 \tag{4.76}$$

Resolvendo a Equação (4.74) com as restrições da Equação (4.76) determina-se que os multiplicadores de Lagrange, α,α^*, com a função de regressão fornecida pela Equação (4.71), em que a predição se torna a Equação (4.77)

$$\bar{\mathbf{w}} = \sum_{i=1}^{l}(\alpha_i-\alpha_i^*)\mathbf{x}_i \qquad \bar{b} = -\frac{1}{2}\langle \bar{\mathbf{w}},(\mathbf{x}_r+\mathbf{x}_s)\rangle \tag{4.77}$$

As condições de Karush-Kuhn-Tucker (KKT) que são satisfeitas pela solução da Equação (4.78) são as seguintes,

$$\bar{\alpha}_i\bar{\alpha}_i^* = 0, \quad i=1,\cdots,l \tag{4.78}$$

Por conseguinte, os vetores de suporte são os pontos onde exatamente um dos multiplicadores de Lagrange seja maior do que zero. Quando $\epsilon = 0$, tem-se a função de perda L_1 e o problema de otimização é simplificado de acordo com a Equação (4.79),

$$\min_{\beta} \frac{1}{2}\sum_{i=1}^{l}\sum_{j=1}^{l}\beta_i\beta_j\langle \mathbf{x}_i, \mathbf{x}_j\rangle - \sum_{i=1}^{l}\beta_i y_i \tag{4.79}$$

com as restrições da Equação (4.80),

$$\begin{gathered} -C \le \beta_i \le C, \quad i = 1, \cdots, l \\ \sum_{i=1}^{l}\beta_i = 0 \end{gathered} \tag{4.80}$$

e a função de regressão dada pela Equação (4.81), em que

$$\begin{gathered} \bar{\mathbf{w}} = \sum_{i=1}^{l}\beta_i \mathbf{x}_i \\ \bar{b} = -\frac{1}{2}\langle \bar{\mathbf{w}}, (\mathbf{x}_r + \mathbf{x}_s)\rangle \end{gathered} \tag{4.81}$$

Função de Perda Quadrática

Utilizando uma função de perdão quadrática, da Figura 4.8(a) (Equação (4.82)),

$$L_{quad}(f(\mathbf{x}) - y) = (f(\mathbf{x}) - y)^2 \tag{4.82}$$

a solução é dada por (Equação (4.83)),

$$\begin{aligned} \max_{\alpha,\alpha^*} \mathbf{W}(\alpha, \alpha^*) = \max_{\alpha,\alpha^*} & -\frac{1}{2}\sum_{i=1}^{l}\sum_{j=1}^{l}(\alpha_i - \alpha_i^*)(\alpha_i - \alpha_i^*)\langle \mathbf{x}_i, \mathbf{x}_j\rangle + \sum_{i=1}^{l}(\alpha_i - \alpha_i^*)y_i \\ & -\frac{1}{2C}\sum_{i=1}^{l}(\alpha_i^2 + (\alpha_i^*)^2) \end{aligned} \tag{4.83}$$

A correspondente otimização pode ser simplificada pela exploração das condições de KKT, da Equação (4.83) e notando que isso implica $\beta_i^* = |\beta_i|$. Os problemas resultantes de otimização são (Equação (4.84)),

$$\min_{\beta} \frac{1}{2}\sum_{i=1}^{l}\sum_{j=1}^{l}\beta_i\beta_j\langle \mathbf{x}_i, \mathbf{x}_j\rangle - \sum_{i=1}^{l}\beta_i y_i + \frac{1}{2C}\sum_{i=1}^{l}\beta_i^2 \tag{4.84}$$

com as restrições(Equação (4.85)),

$$\sum_{i=1}^{l} \beta_i = 0 \tag{4.85}$$

e a função de regressão é dada pelas Equações (4.71) e (4.81).

Função de Perda de Huber

Utilizando a função de perda de Huber, como o da Figura 4.8 (c) (Equação (4.86)),

$$L_{huber}(f(\mathbf{x}) - y) = \begin{cases} \frac{1}{2}(f(\mathbf{x}) - y)^2 & \text{para } |f(\mathbf{x}) - y| < \mu \\ \mu|f(\mathbf{x}) - y| & \text{de outro modo} \end{cases} \tag{4.86}$$

a solução é dada por (Equação (4.87)),

$$\begin{aligned} \max_{\alpha,\alpha^*} \mathbf{W}(\alpha, \alpha^*) = \max_{\alpha,\alpha^*} & -\frac{1}{2}\sum_{i=1}^{l}\sum_{j=1}^{l}(\alpha_i - \alpha_i^*)\left(\alpha_j - \alpha_j^*\right)\langle \mathbf{x}_\mathrm{i}, \mathbf{x}_j\rangle \\ & + \sum_{i=1}^{l}(\alpha_i - \alpha_i^*)y_i - \frac{1}{2C}\sum_{i=1}^{l}(\alpha_i^2 + (\alpha_i^*)^2)\,\mu \end{aligned} \tag{4.87}$$

O resultado do problema de otimização é dado pela Equação (4.88),

$$\min_{\beta} \frac{1}{2}\sum_{i=1}^{l}\sum_{j=1}^{l}\beta_i\beta_j\langle \mathbf{x}_i, \mathbf{x}_j\rangle - \sum_{i=1}^{l}\beta_i y_i + \frac{1}{2C}\sum_{i=1}^{l}\beta_i^2\mu \tag{4.88}$$

Com as restrições apresentadas na Equação (4.89),

$$\begin{aligned} -C \le \beta_i \le C, \quad i = 1, \cdots, l \\ \sum_{i=1}^{l}\beta_i = 0 \end{aligned} \tag{4.89}$$

Regressão Não Linear

De forma similiar aos problemas de classificação, um modelo não linear é geralmente necessário para modelar adequadamente os dados. Da mesma maneira que o SVC não linear, um mapeamento não linear pode ser utilizado para mapear os dados em um espaço característico de alta dimensão onde a regressão linear é realizada. A abordagem de núcleos é novamente

utilizada para tratar sobre a dimensionalidade dos dados de entrada. A solução SVR não linear, utilizando uma função de perda ε-insensitiva, Figura 4.8(d), é dada pela Equação (4.90),

$$\begin{aligned}\max_{\alpha,\alpha^*} \mathbf{W}(\alpha,\alpha^*) = \max_{\alpha,\alpha^*} &\sum_{i=1}^{l} \alpha_i^*(y_i-\varepsilon) - \alpha_i(y_i+\varepsilon) \\ &-\frac{1}{2}\sum_{i=1}^{l}\sum_{j=1}^{l}(\alpha_i-\alpha_i^*)\left(\alpha_j-\alpha_j^*\right)K\left(\mathbf{x}_i,\mathbf{x}_j\right)\end{aligned} \tag{4.90}$$

com as restrições da Equação (4.91),

$$0 \le \alpha_i, \alpha_i^* \le C, \quad i = 1,\cdots,l, \quad \sum_{i=1}^{l}(\alpha_i-\alpha_i^*) = 0 \tag{4.91}$$

Resolvendo a Equação (4.90) com as restrições da Equação (4.91) se determina os multiplicadores de Lagrange α_i, α_i^*, e a função de regressão é dada pela Equação (4.92),

$$f(\mathbf{x}) = \sum_{SVs}(\bar{\alpha}_i - \bar{\alpha}_i^*)K(\mathbf{x}_i,\mathbf{x}) + \bar{b} \tag{4.92}$$

em que (Equação (4.93))

$$\begin{aligned}\langle \bar{\mathbf{w}}, \mathbf{x}\rangle &= \sum_{i=1}^{l}(\alpha_i-\alpha_i^*)K\left(\mathbf{x}_i,\mathbf{x}_j\right) \\ \bar{b} &= -\frac{1}{2}\sum_{i=1}^{l}(\alpha_i-\alpha_i^*)\left(K(\mathbf{x}_i,\mathbf{x}_r)+K(\mathbf{x}_\mathrm{i},\mathbf{x}_s)\right)\end{aligned} \tag{4.93}$$

Assim com o SVC, a restrição de igualdade pode ser descartada se o núcleo contém um termo de polarização, b, sendo acomodado dentro da função núcleo e a função de regressão é dada pela Equação (4.94),

$$f(\mathbf{x}) = \sum_{SVs}(\bar{\alpha}_i - \bar{\alpha}_i^*)K(\mathbf{x}_i,\mathbf{x}) \tag{4.94}$$

Os critérios de otimização para as funções de perda do item *a* são igualmente obtidos substituindo o produto escalar por uma função núcleo. A função de perda ε-insensível é atrativa pois, ao contrário da quadrática e funções de perda de Huber em que todos os pontos de dados

serão vetores de suporte, a solução SV pode ser esparsa. A função de perda quadrática produz uma solução que é equivalente à regressão de cume, ou regularização de ordem 0, em que o parâmetro de regularização é $\lambda = 1/2C$.

Comentários

Nos métodos de regressão é necessário escolher tanto uma função de perda representativa quanto uma capacidade de controle adicional que também pode ser requerida. Essas considerações devem ser baseadas no conhecimento prévio do problema e da distribuição do ruído. Na ausência dessa informação, a função robusta de perda de Huber, Figura 4.8 (c), se apresenta como uma boa alternativa (VAPNIK, 1995). Vapnik desenvolveu uma função de perda ϵ-insensitiva como uma troca entre a função robusta de perda de Huber e uma função que habilita a dispersão entre os SVs. No entanto, a sua implementação é mais onerosa computacionalmente e a região ϵ-insensível pode ter desvantagens.

4.8. Aplicações do SVM em Detecção e Diagnóstico de Falhas

A utilização das Máquinas de Vetores de Suporte para a detecção e diagnóstico de falhas é imediata, já que é baseada nos padrões referentes à operação normal do sistema. Uma vez que o SVM pode ser amplamente utilizado para reconhecimento de padrões, ele também pode ser utilizado para separar padrões diferentes. Através do modelo obtido pelo SVM de classificação ou o SVM de regressão, o controlador seria capaz de detectar (e diagnosticar) padrões que influenciariam de maneira negativa no sistema.

Muitas vezes, é necessário que o SVM seja utilizado de forma a classificar diversos padrões diferentes, sendo impossível utilizar somente um simples SVM classificador entre duas classes. Nesses casos, faz-se necessário utilizar um SVM multicamada capaz de separar os diversos padrões de falhas do padrão de operação normal. Quando as falhas são conhecidas, a detecção (e diagnóstico) se torna praticamente automática, sendo que seria somente necessário realizar o treinamento dessas falhas e da operação normal no SVM classificador. Já no caso de não existir as informações de padrões de falhas, o modelo SVM melhor utilizado seria o SVM de regressão, sendo que este seria capaz de "predizer" a operação normal do sistema, caso o sistema se desviasse desse padrão, seria então detectada então uma falha do processo.

Existem alguns trabalhos importantes na área de detecção de falhas com a utilização de SVM (GHATE; DUDUL, 2009; DEHESTANI *et al.*, 2011; CHEN *et al.*, 2011; SALAHSHOOR *et al.*, 2010; SAMANTARAY *et al.*, 2009). Em muitos deles, o SVM é somente uma técnica aliada ao processo de detecção de falhas, sendo necessário somente para encontrar o modelo que separa as classes de padrões.

Para a utilização de um sistema de detecção de falhas baseado em modelo SVM deve-se seguir as seguintes etapas:

i. Identificar os parâmetros de monitoramento do processo;
ii. Pré-processamento dos dados;
    a. Limpar os dados (remoção de ruídos ou dados inconsistentes);
    b. Transformar os dados (normalização dos dados);
iii. Seleção das características;
    a. Implementar a análise de correlação;
    b. Implementar o algoritmo de decisão em árvore;
    c. Adquirir parâmetros chaves;
iv. Construir o classificador SVM;
    a. Selecionar a função núcleo;
    b. Encontrar a configuração ótima dos parâmetros;
    c. Treinar o classificador SVM;
v. Avaliar o desempenho;

Para utilização do SVM, tanto o de classificação quanto o de regressão, é necessário obter os dados do sistema em questão quando operando normalmente, e, para o SVM de classificação, os dados do sistema em questão quando operando em situações de falha. De posse desses dados, é treinado o SVM para obtenção de um modelo de classificação ou de regressão capaz de realizar a separação dos dados em tempo real, se em operação normal ou em situação de falha.

O sistema de detecção de falhas com SVM apresentado neste trabalho é capaz de diferenciar, instante a instante, se a planta química está na situação de operação normal ou passa por alguma falha.

Neste trabalho, são apresentados três exemplos de utilização do SVM para detecção de falhas, os métodos de máquinas de vetores de suporte de classificação (SVC) *one-against-all* (OAA) e *one-against-one* (OAO) e o método de máquinas de vetores de suporte de regressão (SVR).

4.8.1. Máquinas de vetores de suporte para classificação

A utilização do SVC para detecção e diagnóstico de falhas é baseada na capacidade da técnica em separar classes de dados. De posse dos dados previamente coletados de operações normais e de operações em que ocorreram falhas conhecidas e identificadas pelo controlador,

faz-se o estudo de quais variáveis são importantes para o sistema de detecção, retirando ruído e dados inconsistentes.

Assim que os dados estão pré-tratados, é escolhida qual função núcleo melhor se adapta ao sistema, buscando uma configuração ótima dos parâmetros. Essa configuração dos parâmetros pode ser realizada através de uma simples varredura, procurando no treinamento da SVC obter um modelo que maximize (ou minimize), de acordo com os objetivos da detecção e diagnóstico de falhas, a função custo utilizada. Neste trabalho foi utilizado um algoritmo de evolução diferencial para encontrar os parâmetros ótimos das técnicas SVC.

Após realizado o treinamento das SVC, o modelo obtido será capaz de detectar e diagnosticar em tempo real as falhas de processo treinadas, ou então, dizer se o sistema se encontra em operação normal ou anormal.

One-against-all* e *one-against-one

A máquina de vetor de suporte de classificação multiclasse pode ser obtida através de uma expansão da teoria de classificação binária. O método utilizado primordialmente para a obtenção de classificação de k classes se deu através da obtenção de k modelos SVM de classificação, em que cada um deles classificaria a i-ésima classe e rejeitaria todas as outras classes. Esse método ficou conhecido como *one-against-all* (OAA) ou *one vs. all*.

O método SVC OAA possui claras vantagens na capacidade computacional necessária para a obtenção e treinamento desses k modelos SVM de classificação binária, perante o método utilizado posteriormente, chamado *one-against-one* (OAO) ou *one vs. one*. O método SVC OAO necessita da obtenção de $k(k-1)/2$ modelos SVM de classificação binária, utilizados para categorizar as diferentes classes perante cada uma das outras, tornando o treinamento mais dispendioso, porém, com uma maior precisão.

4.8.2. Máquinas de vetores de suporte para regressão - SVR

A utilização do SVR para detecção de falhas é centrada na capacidade da metodologia em obter modelos de regressão a partir de um número reduzido de dados. Obtidos os dados do sistema, coletados quando em operação normal, é possível treinar o SVR para obter um modelo de predição, que pode apontar ao controlador o momento exato em que o sistema se desviou de sua operação normal.

Após os dados serem tratados, estes devem ser treinados pelo SVR a partir de uma função núcleo escolhida de maneira que esta se adapte melhor aos dados. A configuração dos parâmetros do SVR visa a obtenção de um modelo muito próximo à dinâmica em estado estacionário do processo. Portanto, essa configuração pode novamente ser obtida através de

varredura dos parâmetros ou aplicação de algoritmo capaz de encontrar esses valores ótimos. Foram utilizadas neste trabalho a metodologia de redes neuronais para a obtenção desses parâmetros ótimos.

4.9. Estudo de Caso - Processo de Produção de Ciclopentanol

Nesta seção serão apresentados os resultados do estudo de caso do processo de produção de Ciclopentanol a partir da reação de van der Vusse. A metodologia de simulação e obtenção dos dados do processo foram apresentadas no Capítulo 3.

4.9.1. Máquina de Vetores de Suporte de Classificação *One-Against-One*

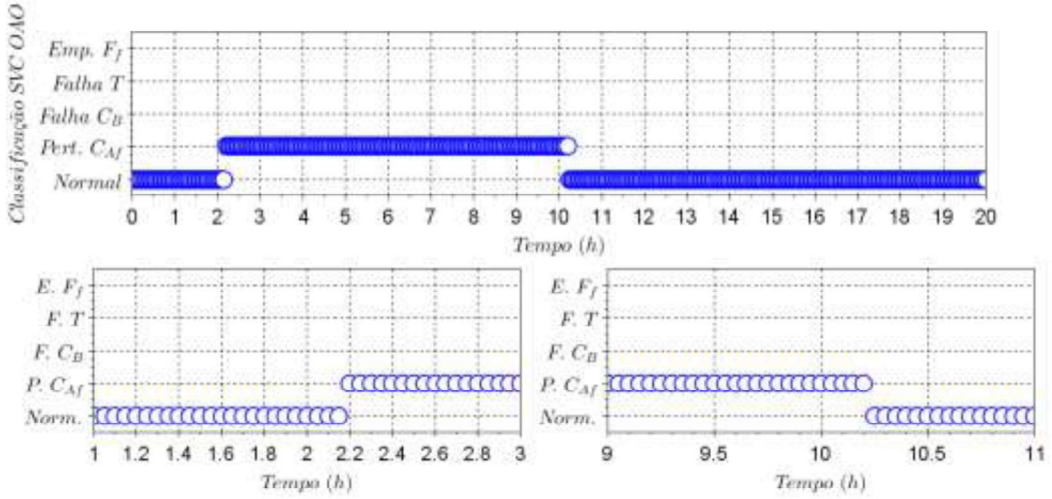
Para a máquina de vetores de suporte de classificação (*Support Vector Classification*, SVC), na fase de treinamento e validação do modelo SVC OAO, é necessário normalizar todos os dados de treinamento referentes ao comportamento normal e aos comportamentos de cada uma das falhas e para cada grupo de dados atribuir um rótulo. Neste caso, foi atribuído rótulo 1 ao comportamento normal, e rótulos 2, 3, 4 e 5, respectivamente para cada uma das falhas (no caso, perturbação em C_{Af}, falha em C_B, falha em T, e emperramento em F_f). Uma vez atribuídos os rótulos, deve-se normalizar todos os dados. Para o SVC, foram utilizadas as variáveis controladas, C_B e T, e as variáveis manipuladas, F_f e $\dot{Q}_W$. Do total de dados, foram utilizados em torno de dois terços dos dados simulados para o treinamento e o restante, um terço, foi utilizado para a validação do modelo de acordo com a metodologia apresentada no Capítulo 3.

O sistema de detecção através do SVC utiliza as informações treinadas para categorizar as falhas e a operação normal. Independentemente do instante em que a falha seja aplicada, o sistema de detecção deve ter a capacidade de categorizar cada instante da operação do processo.

Uma vez obtido o modelo SVC OAO com precisão de 98,797%, quando comparados os vetores de saída dos dados de validação com os dados do modelo treinado, identificaram-se os sinais simulados para a detecção e diagnóstico de falhas. O SVC OAO separou 148 vetores de suporte do total de 1336 dados de treinamento, bem como os respectivos vetores de suporte de classificação *one-against-one* das cinco classes (quatro falhas e uma operação normal). Foram definidos os melhores valores de C para cada modelo SVC OAO treinado.

A Figura 4.9 apresenta os rótulos obtidos através do modelo SVC, da Perturbação C_{Af}, aplicada no instante 2 h e retirada no instante 10 h, em que o controlador PI foi capaz de rejeitar a perturbação.

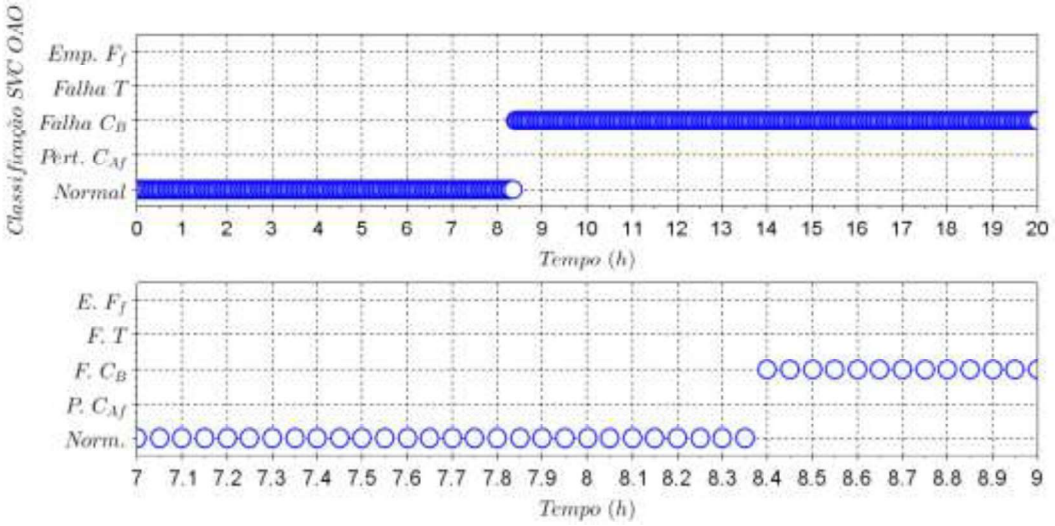
Figura 4.9 - Rótulos obtidos através do processo de detecção da perturbação C_{Af} – SVC OAO.



O treinamento do modelo SVC foi realizado de forma que o sistema de detecção e diagnóstico de falhas fosse capaz de identificar a perturbação do instante em que foi aplicada até o momento em que esta cessasse. A perturbação levou 3 intervalos de amostragem (0,15 h) para ser detectada, e, posteriormente, levou 4 intervalos de amostragem (0,20 h) para o sistema de detecção identificar novamente a operação normal.

A Figura 4.10 apresenta os rótulos obtidos através do modelo SVC, da Falha C_B aplicada no instante 8 h. A falha foi detectada após 7 intervalos de amostragem (0,35 h). Apesar do tempo considerável para a detecção da falha, pode-se estabelecer que o sistema de detecção e diagnóstico de falhas cumpriu o proposto, realizando a detecção da falha.

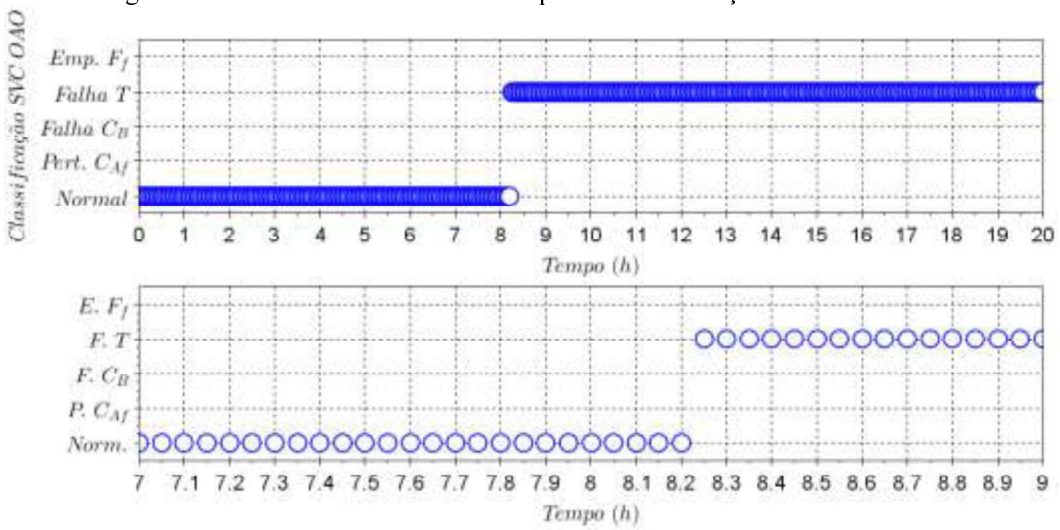
Figura 4.10 - Rótulos obtidos através do processo de detecção da falha C_B - SVC.



A Figura 4.11 apresenta os rótulos obtidos através do modelo SVC, da Falha *T* aplicada no instante 8 h. A falha foi detectada após 4 intervalos de amostragem (0,20 h). Apesar do

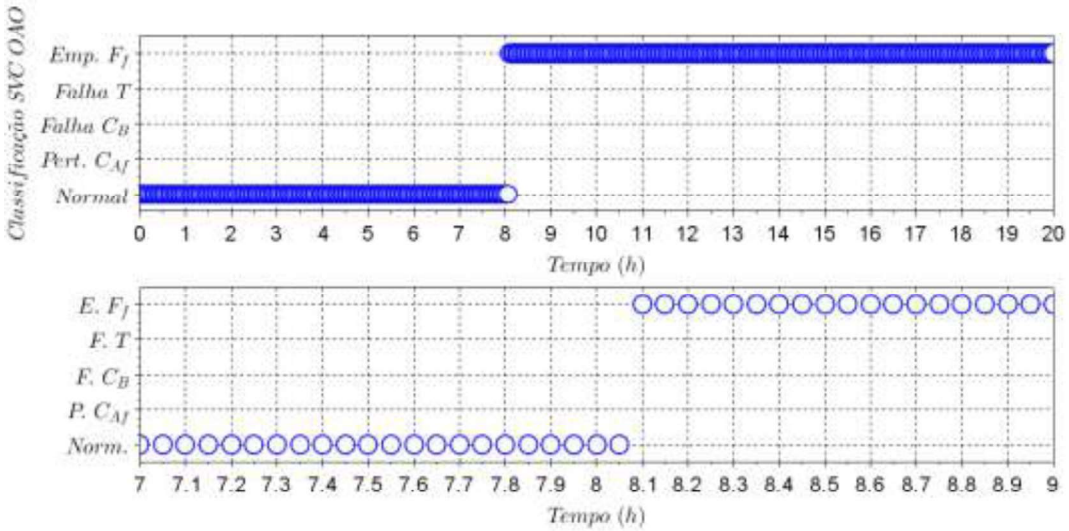
tempo considerável para a detecção da falha, pode-se estabelecer que o sistema de detecção e diagnóstico de falhas cumpriu o proposto, realizando a detecção da falha.

Figura 4.11 - Rótulos obtidos através do processo de detecção da falha *T* - SVC.



A Figura 4.12 apresenta os rótulos obtidos através do modelo SVC, do emperramento F_f aplicada no instante 8 h. A falha foi detectada após 1 intervalo de amostragem (0,05 h). Os resultados obtidos para o Emperramento F_f são ótimos, uma vez que a falha foi detectada quase que instantaneamente, somente após um intervalo de amostragem. Além disso, não foi detectado erro de classificação.

Figura 4.12 - Rótulos obtidos através do processo de detecção do emperramento F_f - SVC.



Foram contabilizados todos os instantes nos quais o comportamento do sistema era normal e todos os instantes em que o sistema estava passando pelas diferentes falhas treinadas de maneira que foram criados índices que apresentam qual a porcentagem de identificações

corretas ou não, de acordo com a Equação (3.1). A Tabela 4.1 apresenta os índices encontrados para o sistema de detecção e diagnóstico através do SVC OAO.

Tabela 4.1 - Índices encontrados - SVC *one-against-one*.

Oper.	Índices - SVC *one-against-one*			
	DF/F	DF/N	DN/N	DN/F
Pert. C_{Af}	0,9813	0,0167	0,9833	0,0187
Falha C_B	0,9708	0,00	1,00	0,0292
Falha T	0,9833	0,00	1,00	0,0167
Emp. F_f	0,9958	0,00	1,00	0,0042

Os resultados apresentados na Tabela 4.1 demonstram que o SVC OAO tem a capacidade de reconhecer as diferentes classes de falhas, mesmo que a detecção ocorra após alguns instantes. Em nenhum dos quatro sinais com as falhas propostas ocorreram classificações errôneas, somente um *delay* na detecção das falhas supracitadas, quanto ao início das falhas e ao fim da falha no caso da perturbação C_{Af}. A detecção da operação normal em todas as situações foi rápida ocorrendo poucos diagnósticos errados entre a operação normal e as falhas treinadas.

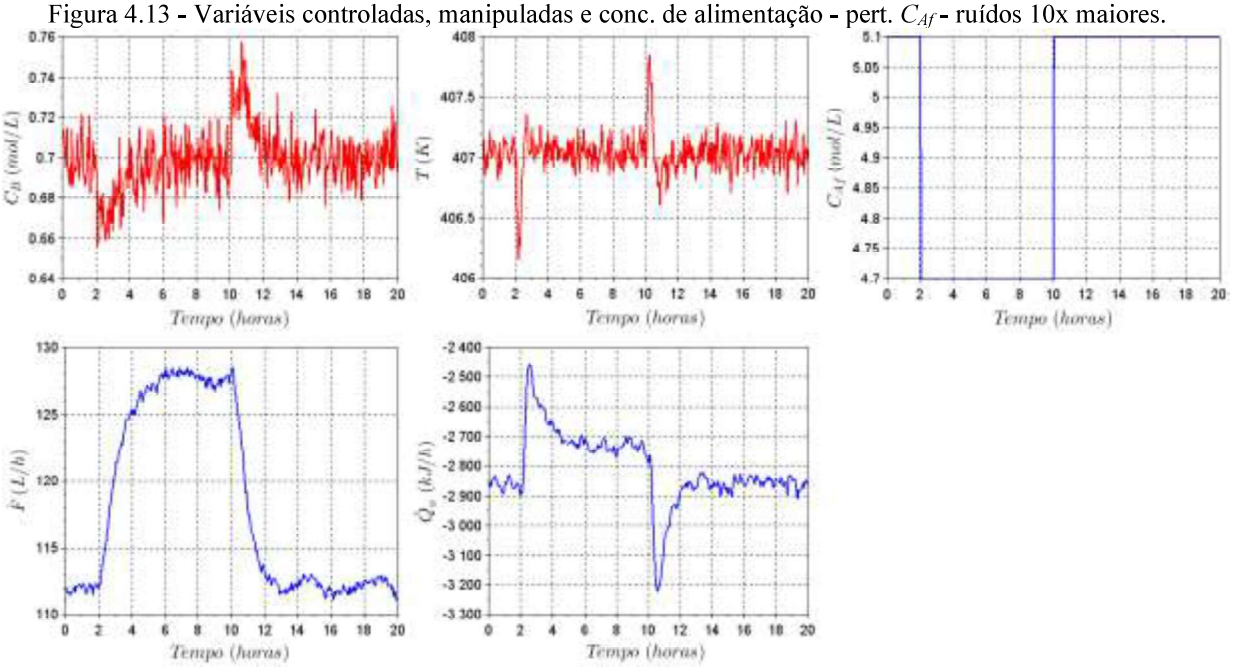
Para efeito de comparação, foi realizada também a adequação da metodologia SVC OAO em sistemas cujo ruído das variáveis amostradas fosse diferente do ruído existente nas variáveis treinadas.

Avaliação de amostras com ruído de maior intensidade - SVC OAO

Para confirmar a eficácia do modelo de detecção e diagnóstico de falhas através do SVC *one-against-one*, realizou-se a detecção de falhas em dados que apresentavam maior ruído do que os dados utilizados no treinamento do sistema. Estabeleceu-se ruídos dez vezes maiores para todas as variáveis simuladas, foi selecionado um ruído aleatório com distribuição gaussiana de média 0 e variância 10^{-4} para a concentração C_B, enquanto que para a temperatura do reator T, para a vazão de alimentação do reator F_f e para quantidade de energia removida na camisa do reator $\dot{Q}_W$, foram selecionados ruídos aleatórios com distribuição gaussiana de média 0 e variância 10^{-2}.

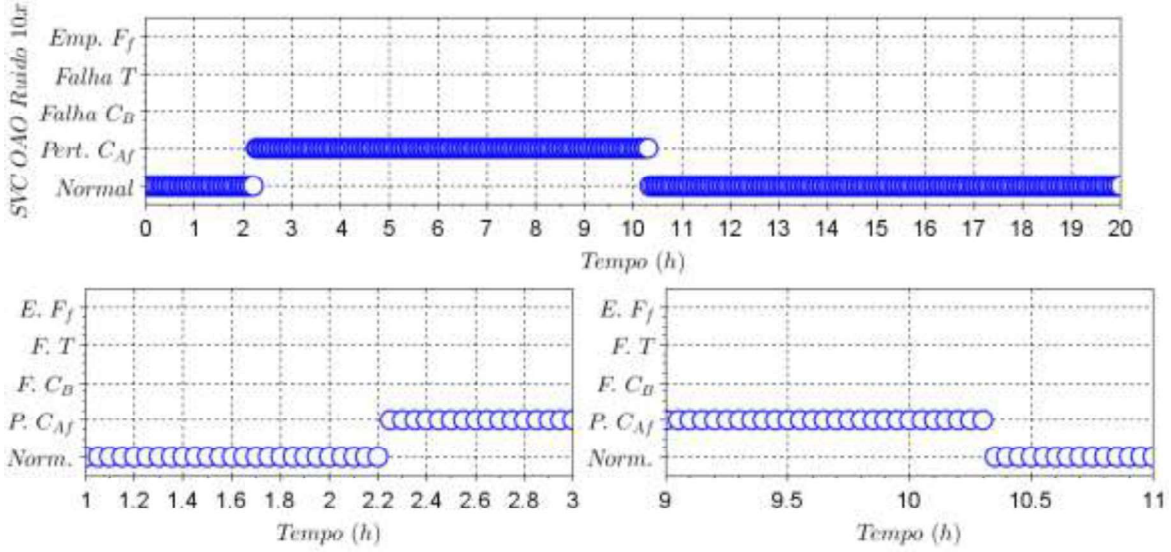
A Figura 4.13 apresenta os dados para a perturbação de C_{Af} com ruídos dez vezes maiores.

Figura 4.13 - Variáveis controladas, manipuladas e conc. de alimentação - pert. C_{Af} - ruídos 10x maiores.



O resultado para a detecção e diagnóstico do sinal da perturbação C_{Af}, com ruído dez vezes maiores, utilizando o método SVC *one-against-one* é apresentado na Figura 4.14.

Figura 4.14 - Rótulos obtidos através do processo de detecção da pert. C_{Af} - SVC OAO - ruídos 10x maiores.

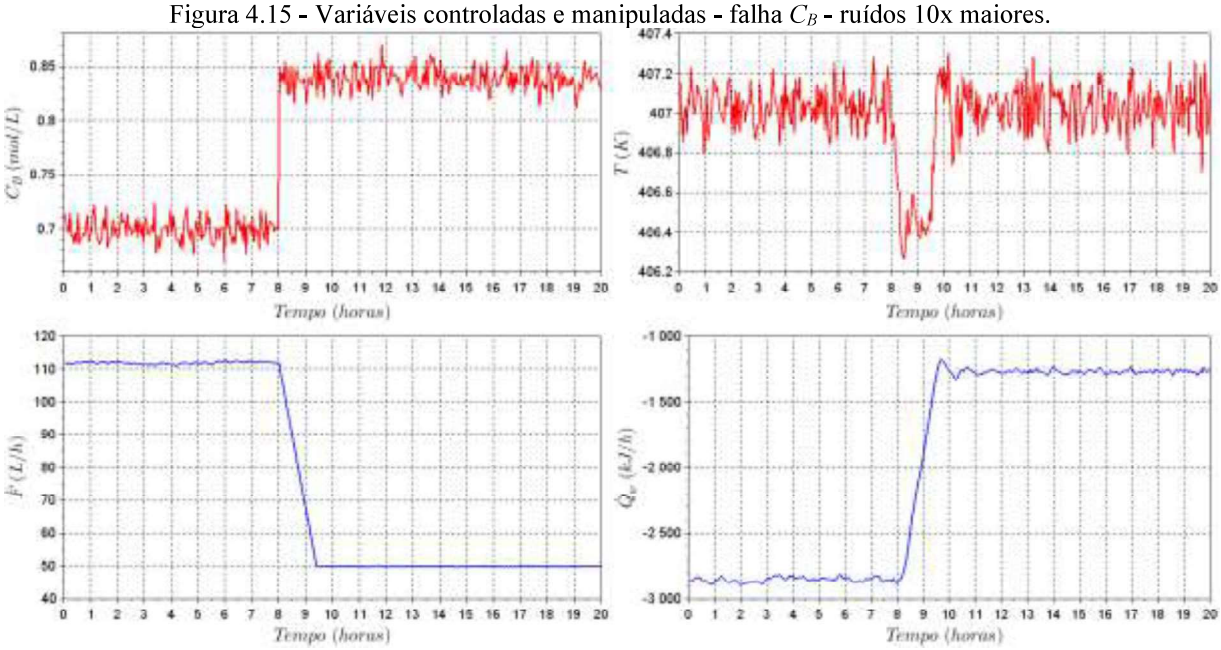


O comportamento do sistema de detecção de falhas para a perturbação C_{Af}, quando classificados os dados com ruídos dez vezes maiores, demonstrou pequena alteração em relação ao padrão observado anteriormente quando classificados dados com ruídos menores detectando a perturbação C_{Af} após 4 intervalos de amostragem, ou 0,20 h (com ruídos menores foi detectada a perturbação após 3 intervalos de amostragem, ou 0,15 h). A operação normal foi detectada após passados 6 intervalos de amostragem quando do fim da perturbação no instante 10 h, ou

0,30 h após o instante 10 h (com ruídos menores foi detectada a operação normal após 4 intervalos de amostragem, ou 0,20 h).

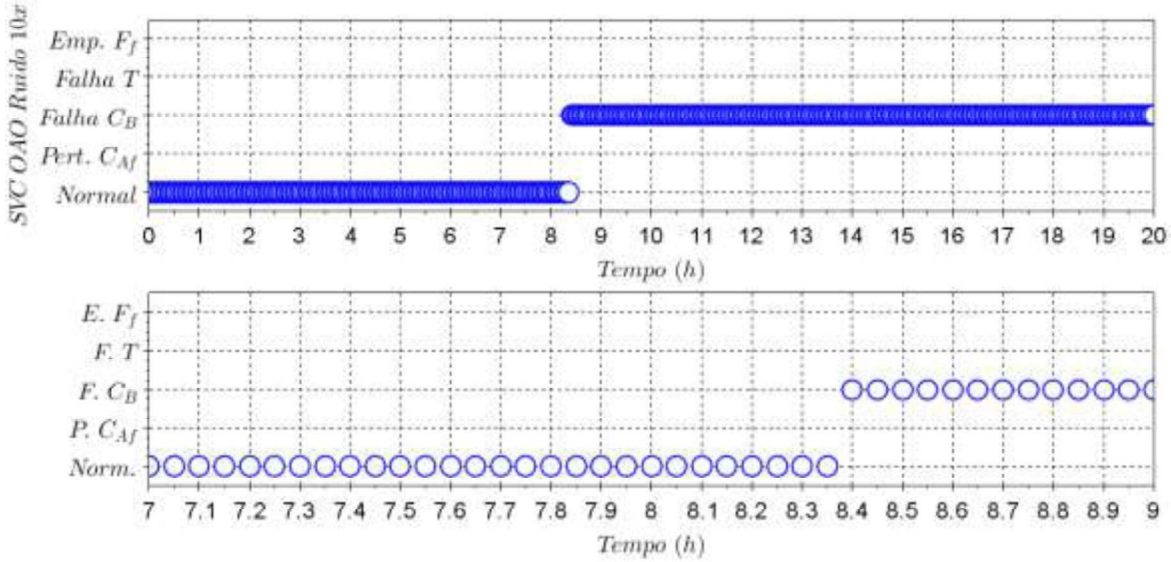
O comportamento dos sinais para a Falha C_B com ruído aleatório 10 vezes maior é apresentado na Figura 4.15.

Figura 4.15 - Variáveis controladas e manipuladas - falha C_B - ruídos 10x maiores.



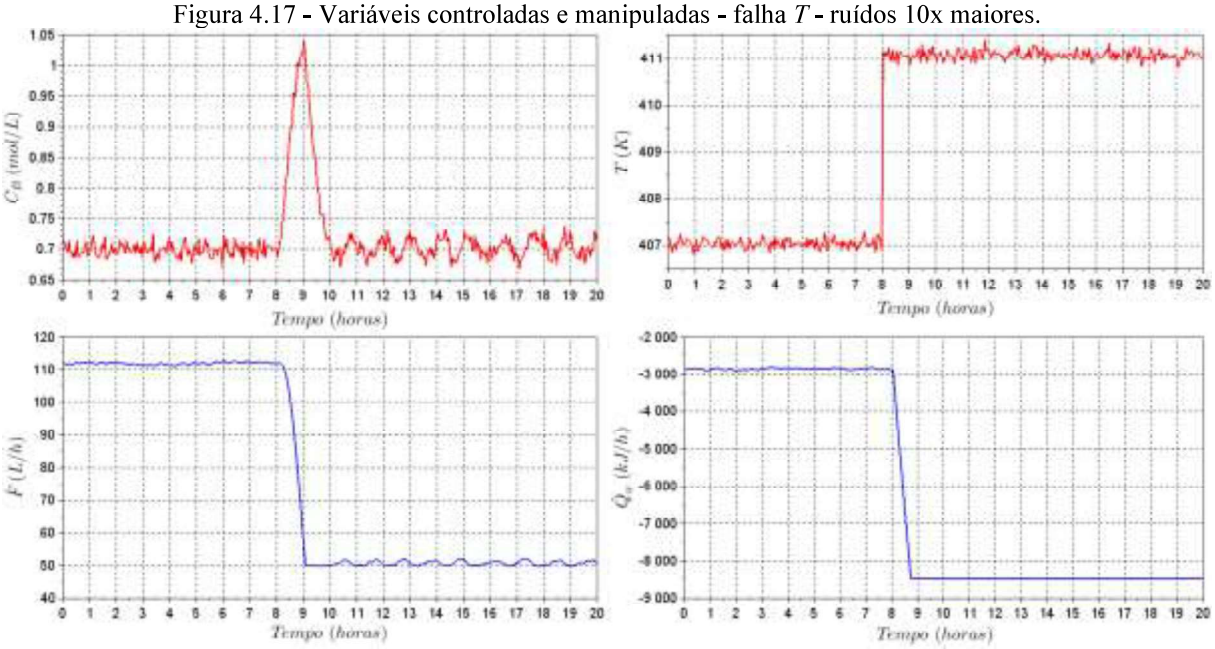
O resultado para a detecção e diagnóstico para o sinal da Falha C_B com ruído 10 vezes maior, utilizando o método SVC OAO é apresentado na Figura 4.16. O comportamento do sistema de detecção de falhas para a falha C_B, quando classificados os dados com ruídos dez vezes maiores, foi o mesmo em relação ao padrão observado quando classificados dados com ruídos menores, detectando a falha C_B após 7 intervalos de amostragem, ou 0,35 h.

Figura 4.16 - Rótulos obtidos através do processo de detecção da falha C_B - SVC OAO - ruídos 10x maiores.



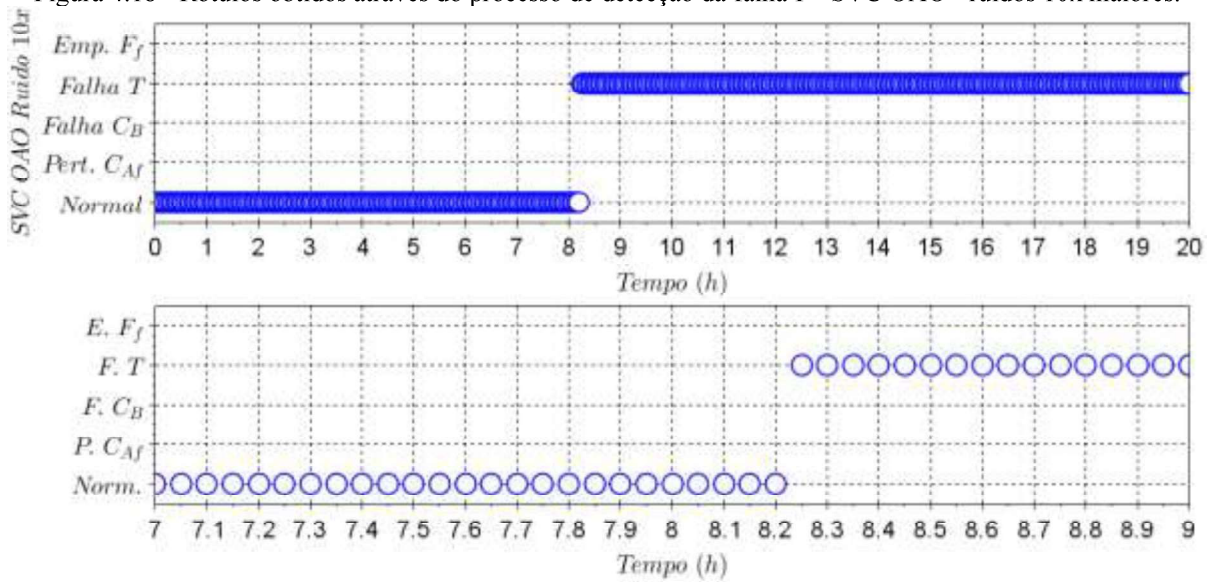
Os sinais da Falha *T* com ruído aleatório 10 vezes maior são apresentados na Figura 4.17. Lembrando que, o ruído gaussiano aleatório em cada uma das variáveis foi dez vezes maior, para a concentração foi de média 0 e variância 10^{-4} e para as outras variáveis foi de média 0 e variância 10^{-2}.

Figura 4.17 - Variáveis controladas e manipuladas - falha *T* - ruídos 10x maiores.



O resultado para a detecção e diagnóstico para o sinal da Falha *T* com ruído 10 vezes maior, utilizando o método SVC OAO é apresentado na Figura 4.18.

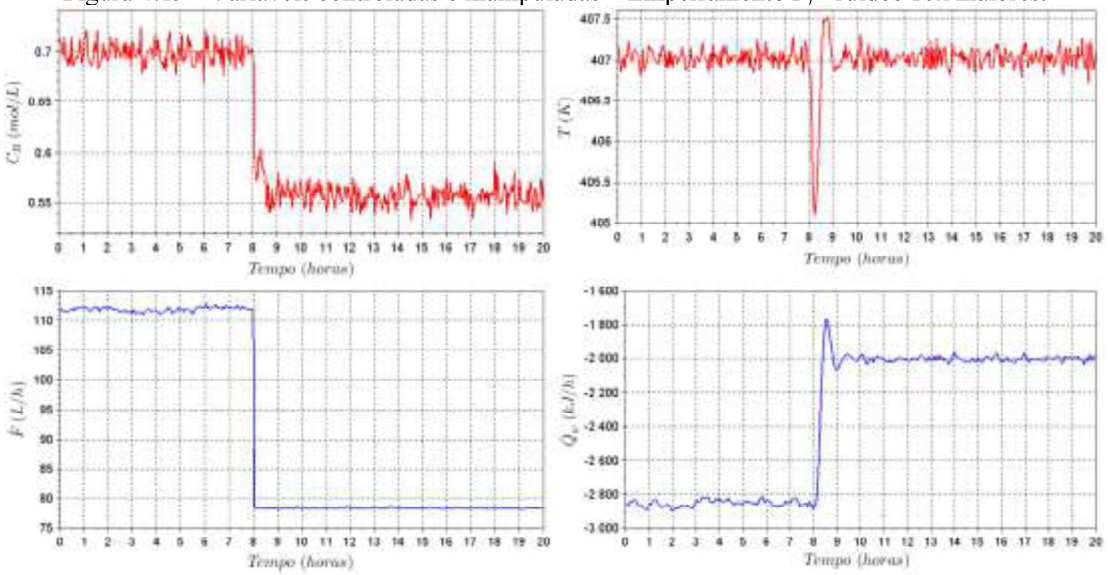
Figura 4.18 - Rótulos obtidos através do processo de detecção da falha *T* - SVC OAO - ruídos 10x maiores.



O comportamento do sistema de detecção de falhas para a falha *T*, quando classificados os dados com ruídos dez vezes maiores, foi o mesmo em relação ao padrão observado quando classificados dados com ruídos menores, detectando a falha *T* após 4 intervalos de amostragem, ou 0,20 h.

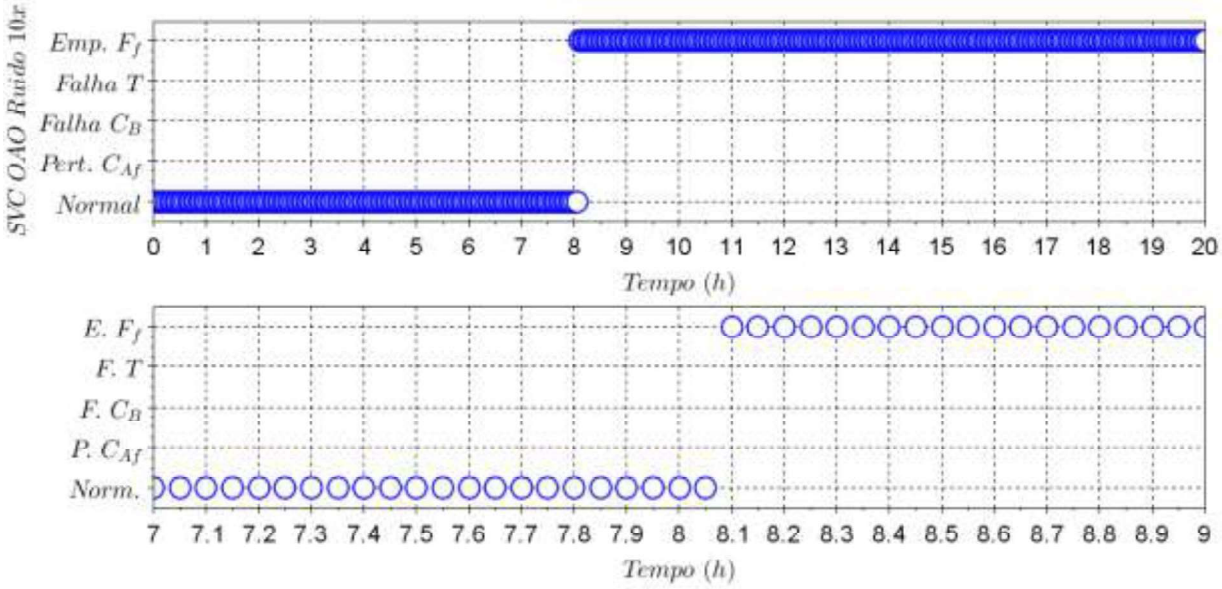
O comportamento dos sinais para o Emperramento F_f com ruído aleatório dez vezes maior é apresentado na Figura 4.19.

Figura 4.19 - Variáveis controladas e manipuladas – Emperramento F_f - ruídos 10x maiores.



O resultado para a detecção e diagnóstico para o sinal do Emperramento *F* com ruído 10 vezes maior, utilizando o método SVC OAO é apresentado na Figura 4.20.

Figura 4.20 - Rótulos obtidos através do processo de detecção da Emp. *F* - SVC OAO - ruídos 10x maiores.



Apesar do ruído ter influência na variação instantânea do sinal do processo, o sistema de detecção de falhas foi capaz de identificar do mesmo modo o Emperramento F_f, independentemente da frequência do ruído amostrado, detectando o emperramento F_f após 1 intervalo de amostragem, ou 0,05 h.

As Figuras Figura 4.14, Figura 4.16, Figura 4.18 e Figura 4.20 confirmam a eficácia do método SVC OAO para detecção e diagnóstico de falhas mesmo com dados amostrados com ruídos maiores que dos dados de treinamento. A Tabela 4.2 apresenta os índices encontrados para o sistema de detecção e diagnóstico através do SVC OAO.

Tabela 4.2 - Índices encontrados - SVC OAO - ruídos 10x maiores.

Oper.	Índices - SVC OAO - ruídos 10x maiores			
	DF/F	DF/N	DN/N	DN/F
Pert. C_{Af}	0,9750	0,0250	0,9750	0,0250
Falha C_B	0,9708	0,00	1,00	0,0292
Falha T	0,9833	0,00	1,00	0,0167
Emp. F_f	0,9958	0,00	1,00	0,0042

Analisando a Tabela 4.2, percebe-se que os índices obtidos apresentam uma situação de igualdade em relação a utilização de ruídos menores, com exceção da perturbação C_{Af}. Os valores obtidos através da análise dos índices, quando utilizados dados amostrais com ruído dez vezes maiores. foram os mesmos apresentados na Tabela 4.1. Esse comportamento mostra que o SVC OAO utilizado para detecção e diagnóstico de falhas é pouco sensível a ruídos em dados amostrais.

4.9.2. Máquina de Vetores de Suporte de Classificação *One-Against-All*

Para a máquina de vetores de suporte de classificação na fase de treinamento e validação do modelo SVC *one-against-all* ou *one vs. all* (SVC OAA), da mesma forma que para o SVC OAO, é necessário normalizar todos os dados de treinamento referentes ao comportamento normal e aos comportamentos de cada uma das falhas atribuindo um rótulo para cada grupo de dados.

Foram atribuídos rótulos de acordo com os comportamentos do sistema: normal (1) e para cada uma das falhas, Perturbação C_{Af} (2), Falha C_B (3), Falha T (4), e Emperramento F_f (5). Para o SVC OAA, também foram utilizadas as variáveis controladas, C_B e T, e as variáveis manipuladas, F_f e $\dot{Q}_w$, para treinamento do sistema de detecção e diagnóstico.

O sistema de detecção através do SVC OAA utiliza as informações treinadas para categorizar as falhas e a operação normal utilizando k modelos SVM, em que a classe i é classificada (y = +1) enquanto todas as outras classes diferentes de i são rejeitadas (y = -1).

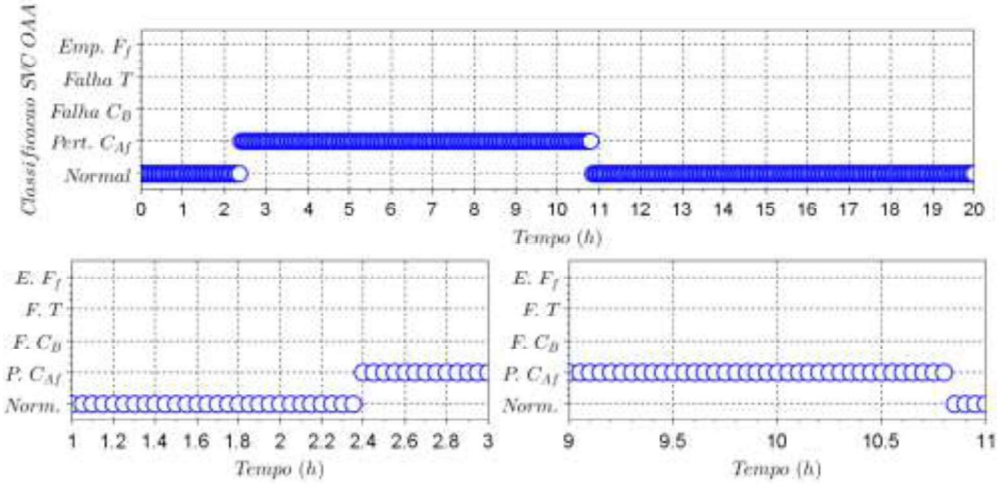
Independentemente do instante em que a falha seja aplicada, o sistema de detecção deve ter a capacidade de categorizar cada instante da operação do processo.

Dos dados utilizados para treinamento e validação do SVC OAA de todas as operações (normal e falhas), são separados dois terços dos dados para treinamento e um terço para a validação, sendo selecionados um dado de validação a cada dois dados de treinamento utilizados.

Uma vez obtido o modelo SVC OAA com precisão de 98,2% dos dados de validação perante os dados de treinamento, foi realizada a identificação dos diferentes sinais para a detecção e diagnóstico de falhas de acordo com as Figuras Figura 3.2 a Figura 3.6. Juntamente com essa análise, foram obtidos os valores dos índices definidos na Equação (3.1).

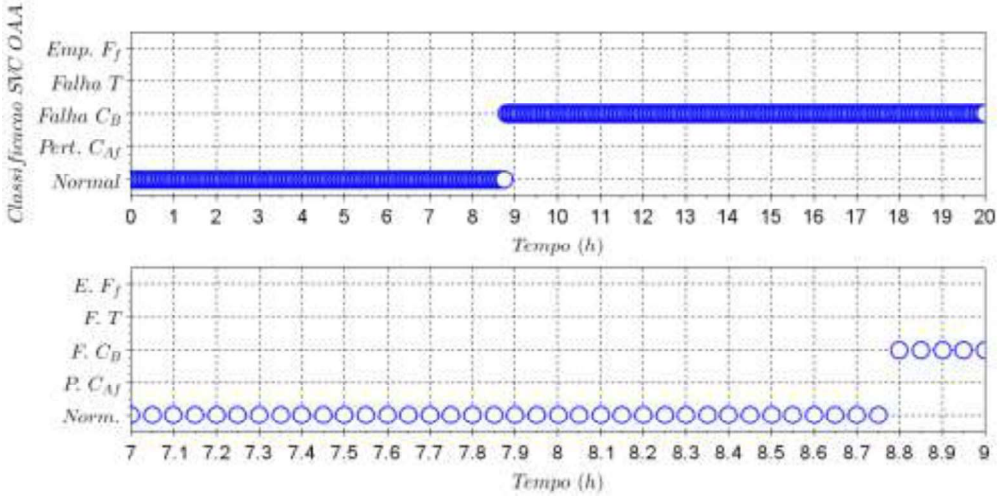
A Figura 4.21 apresenta os rótulos obtidos através do modelo SVC, da Perturbação C_{Af} aplicada no instante 2 h e retirada no instante 10 h. A perturbação C_{Af} foi detectada após 7 intervalos de amostragem (0,35 h). O sistema de detecção e diagnóstico acusou o retorno do sistema a operação normal após 16 intervalos de amostragem (0,8 h) que a perturbação da variável C_{Af} foi detectada. Esse intervalo foi considerado longo e, em alguns casos, pode causar prejuízo às ações a serem tomadas pelos operadores da planta química perante a informação obtida pelo sistema de detecção e diagnóstico de falhas.

Figura 4.21 - Rótulos obtidos através do processo de detecção da pert. C_{Af} - SVC OAA.



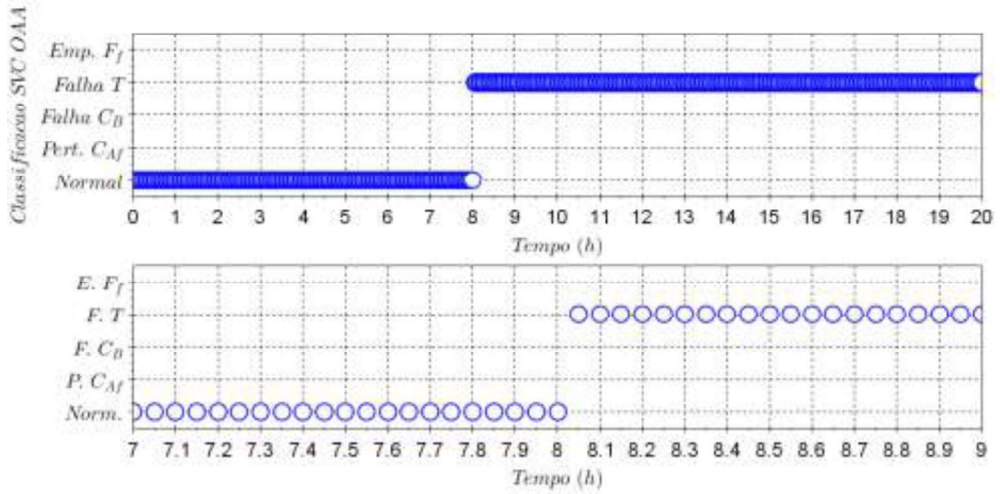
A Figura 4.22 apresenta os rótulos obtidos através do modelo SVC OAA da Falha C_B aplicada no instante 8 h. A Falha C_B foi detectada pelo sistema de detecção e diagnóstico de falhas após 15 intervalos de amostragem (0,75 h).

Figura 4.22 - Rótulos obtidos através do processo de detecção da falha C_B - SVC OAA.



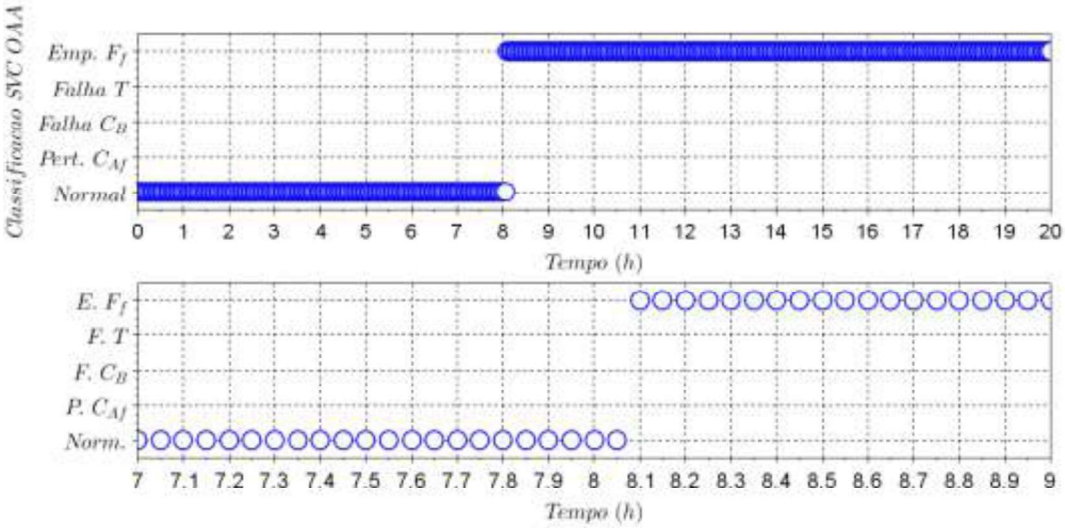
A Figura 4.23 apresenta os rótulos obtidos através do modelo SVC OAA da falha *T* aplicada no instante 8 *h*. A detecção ocorreu imediatamente após a aplicação da falha.

Figura 4.23 - Rótulos obtidos através do processo de detecção da falha *T* - SVC OAA.



A Figura 4.24 apresenta os rótulos obtidos através do modelo SVC OAA do emperramento *F* aplicada no instante 8 *h*. A falha passou a ser detectada após 1 intervalo de amostragem (0,05 h).

Figura 4.24 - Rótulos obtidos através do processo de detecção do Emp. *F* - SVC OAA.



Foram contabilizados todos os instantes nos quais o comportamento do sistema era normal e todos os instantes em que o sistema estava passando pelas diferentes falhas treinadas, de forma que foram criados índices que mostram qual a porcentagem de identificações corretas, ou não, de acordo com as Equações (3.1) e (3.2) para todas as operações com falhas testadas com o sistema de detecção e diagnóstico SVC OAA.

A Tabela 4.3 apresenta os índices encontrados para o sistema de detecção e diagnóstico através do SVC OAA.

Tabela 4.3 - Índices encontrados - SVC OAA.

Oper.	Índices - SVC OAA			
	DF/F	DF/N	DN/N	DN/F
Pert. C_{Af}	0,95625	0,0667	0,8423	0,04375
Falha C_B	0,93750	0,00	1,00	0,06250
Falha T	1,00	0,00	1,00	0,00
Emp. F_f	0,99583	0,00	1,00	0,00417

Comparando os métodos SVC OAO e SVC OAA, através da Tabela 4.1 e da Tabela 4.3, o método SVC OAA obteve bons resultados para duas das quatro operações analisadas (falha T e emperramento F_f), enquanto que o SVC OAO realizou melhores detecções da perturbação C_{Af} e da falha C_B.

Comparando o método SVC OAO com o método SVC OAA, a capacidade computacional envolvida na obtenção dos $k = 5$ modelos SVC OAA é menor do que na obtenção dos $k(k-1)/2 = 10$ modelos envolvidos na detecção através do SVC OAO. O modelo SVC OAA apresentou certo *lag* (atraso) na detecção de duas das falhas, perturbação C_{Af} e falha C_B. Ambos os métodos não apresentaram classificações de falhas diferentes da falha aplicada, demonstrando eficiência dos métodos para a detecção de padrões.

4.9.3. Máquina de Vetores de Suporte de Regressão

Para os métodos de detecção de falhas através de sistemas de regressão, apresenta-se dois métodos: as máquinas de vetores de suporte de regressão e as redes neuronais artificiais de regressão.

Para realizar a detecção de falhas utilizando a metodologia de máquina de vetores de suporte de regressão (SVR), foi realizada a modelagem da regressão através do SVM fornecendo como variáveis de entrada os comportamentos das variáveis manipuladas ($\dot{F}$ e $\dot{Q}_w$) e controladas (C_B e T), de acordo com a Equação (4.95), como um vetor coluna **x**.

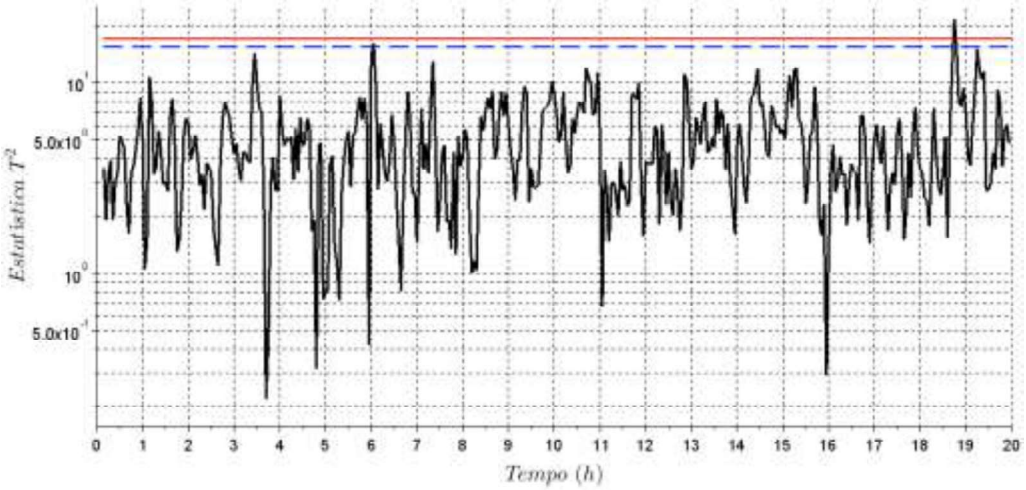
$$\begin{aligned}\mathbf{x} = [&F_f(k)\ \dot{Q}_w(k)\ C_B(k-1)\ T(k-1)\ F_f(k-1)\ \dot{Q}_w(k-1)\ \dots \\ &\dots\ C_B(k-p)\ T(k-p)\ F_f(k-p)\ \dot{Q}_w(k-p)]^{\mathrm{T}}\end{aligned} \tag{4.95}$$

em que k são os instantes no presente, e $k-p$ são p instantes de amostragem no passado.

Com os instantes anteriores, o modelo SVR deve ser capaz de encontrar o valor da variável no instante atual. Para este estudo de caso, foram utilizados 2 instantes no passado (p = 2), de acordo com uma avaliação prévia do comportamento das variáveis. A saída do sistema de detecção é definida como o valor da variável C_B no instante k, ou seja, no presente. O sinal de C_B obtido através do modelo SVR para a operação normal é comparado ao sinal do respectivo comportamento da variável C_B de cada conjunto de dados compostos por operações faltosas e operação normal.

A Figura 4.25 apresenta o comportamento dos valores da estatística multivariada T^2 de Hotelling utilizando o SVR do sinal da Figura 3.6, ou seja, da operação normal. Em todos os gráficos da estatística T^2 de Hotelling, a reta tracejada curta é o alarme do sistema de detecção, enquanto que a reta contínua é o limite de não detecção do sistema.

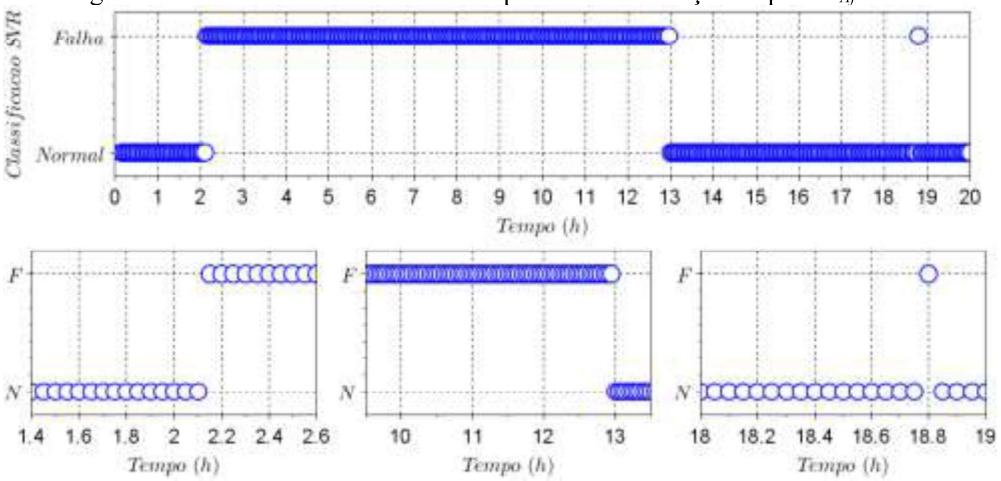
Figura 4.25 - Estatística T² - Operação normal - SVR.



Para a obtenção do modelo SVR da operação normal, foram utilizados dois terços dos dados para treinamento e um terço dos dados para a validação, dentre estes um ponto amostral a cada dois pontos amostrais selecionados para o treinamento. Uma vez obtido o modelo SVR com precisão de 99,5% em relação aos dados de validação, foi realizada a identificação do sinal para a detecção de falhas.

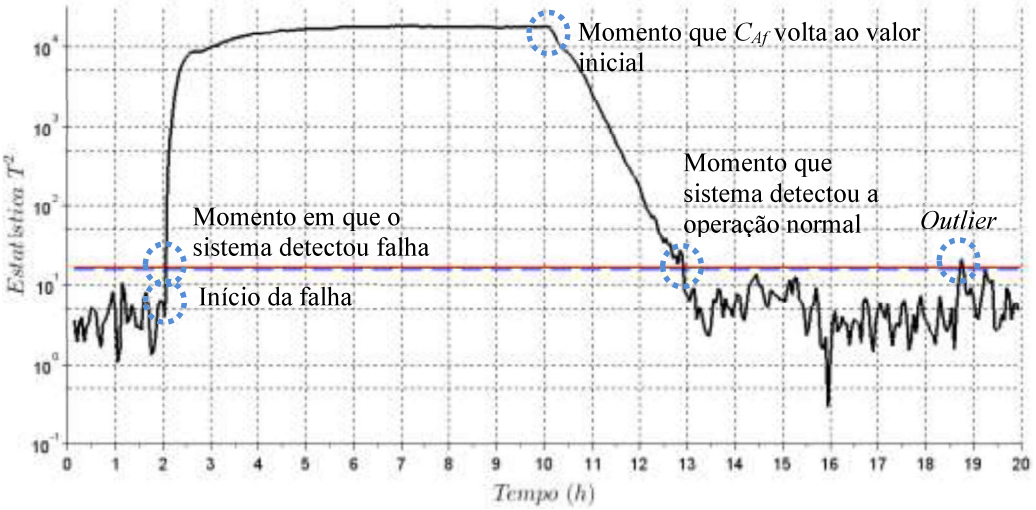
A Figura 4.26 apresenta os rótulos obtidos através do modelo SVR para o sinal do sistema passando pela Perturbação C_{Af} do instante 2 h ao instante 10 h. No SVR, por ser um modelo de regressão, o sistema de detecção de falhas só é capaz de detectar as operações diferentes da operação normal.

Figura 4.26 - Rótulos obtidos através do processo de detecção da pert. C_{Af} - SVR.



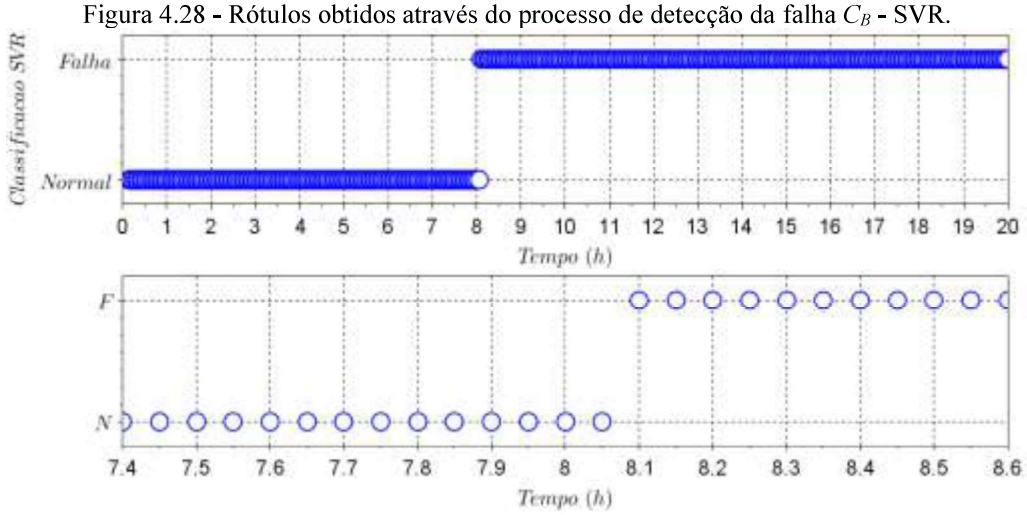
O comportamento do sistema de detecção de falhas através do SVR apresentado na Figura 4.26 mostra que a detecção da perturbação aconteceu após 2 intervalos de amostragem (0,10 h), porém, o sistema não foi rápido para voltar à operação normal após a remoção da perturbação em C_{Af}. O sistema levou em torno de 60 intervalos de amostragem (3,0 h) para retomar a detecção da operação normal. No instante 18,8 h houve a detecção de falha enquanto o sistema se encontrava em operação normal. A Estatística T^2 de Hotelling é apresentada para a confirmar a detecção da modificação de operação, como apresentado na Figura 4.27.

Figura 4.27 - Estatística T² - perturbação C_{Af} - SVR.



A Figura 4.28 apresenta os rótulos obtidos através do sistema de detecção utilizando o modelo SVR para o sinal do sistema reacional passando pela Falha C_B a partir do instante 8 h. O modelo de detecção levou 1 intervalo de amostragem (0,05 h) para detectar que o sistema transitou de uma operação normal para uma operação faltosa.

Figura 4.28 - Rótulos obtidos através do processo de detecção da falha C_B - SVR.



A Estatística T² de Hotelling foi utilizada para monitorar a modificação de operação normal para a operação faltosa, como apresentado na Figura 4.29.

A Figura 4.30 apresenta os rótulos obtidos através do sistema de detecção utilizando o modelo SVR para o sinal do sistema reacional passando pela Falha *T* a partir do instante 8 h. O modelo SVR levou 1 intervalo de amostragem (0,05 h) para detectar que o processo passou de uma operação normal para uma operação faltosa.

Figura 4.29 - Estatística T² - Falha C_B - SVR.

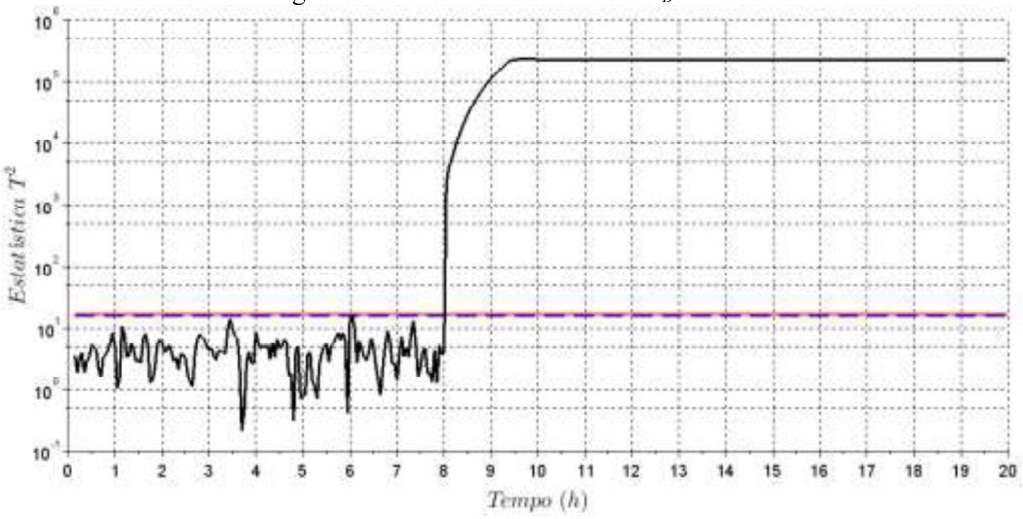
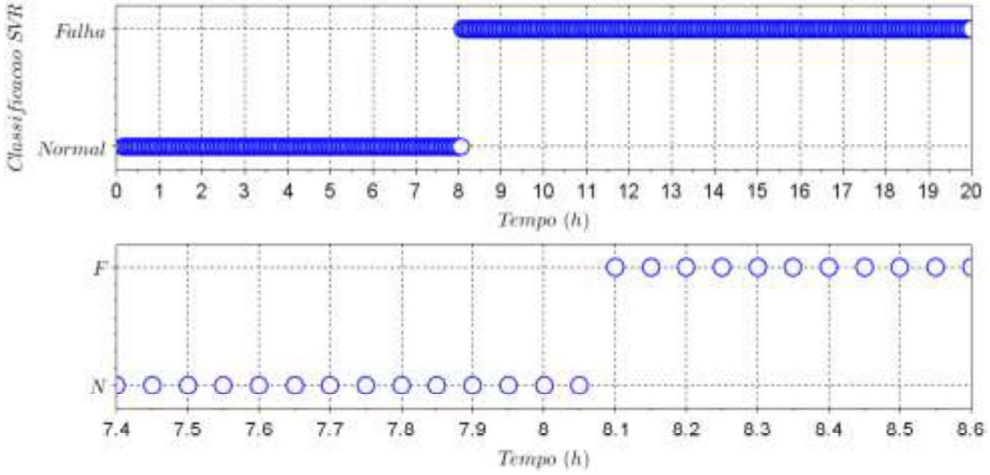
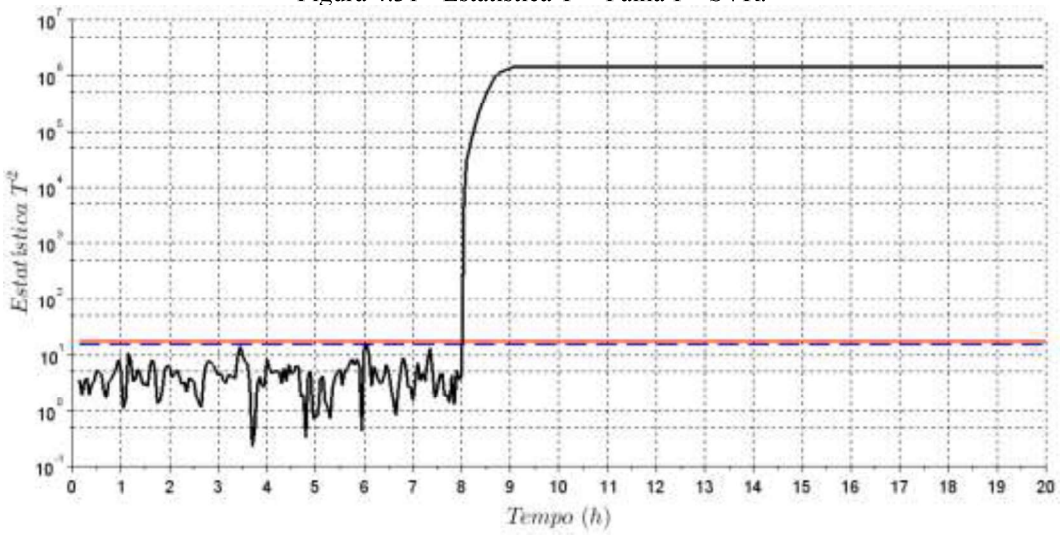


Figura 4.30 - Rótulos obtidos através do processo de detecção da falha T - SVR.



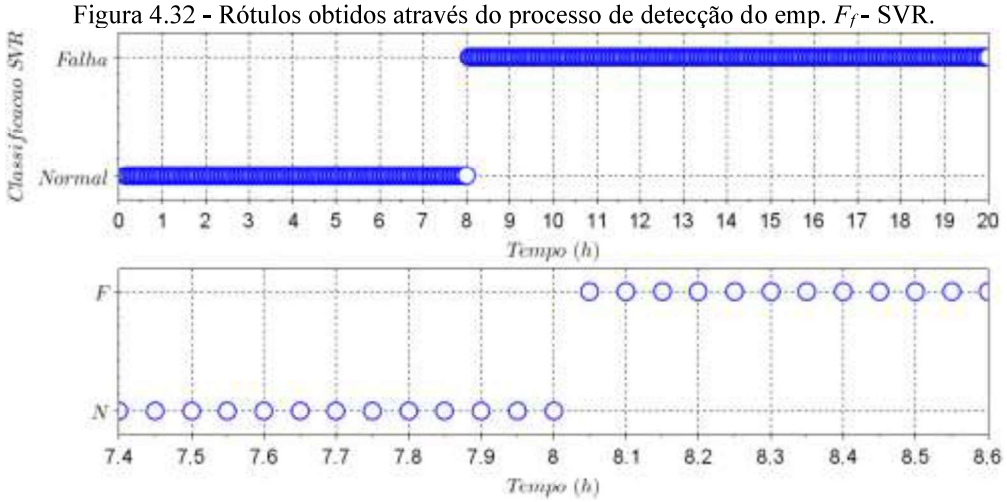
A Estatística T² de Hotelling foi utilizada para monitorar a modificação da operação normal para operação faltosa, como apresentado na Figura 4.31.

Figura 4.31 - Estatística T² - Falha T - SVR.



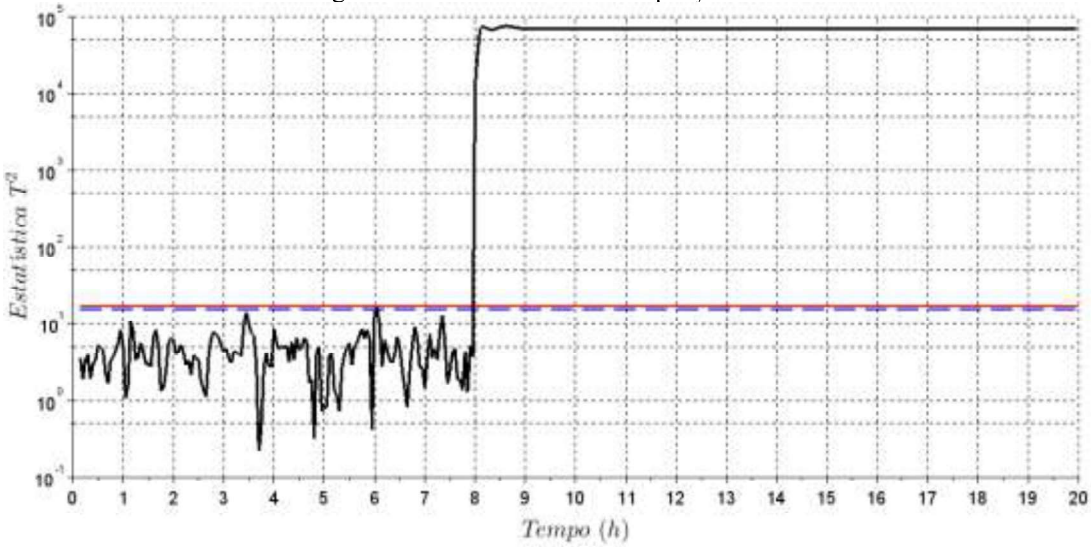
A Figura 4.32 apresenta os rótulos obtidos através do sistema de detecção utilizando o modelo SVR para o sinal do sistema reacional passando pelo Emperramento de F_f a partir do instante 8 h. O modelo SVR detectou a modificação de operação de normal para falha instantaneamente após aplicada a falha. O sistema de detecção SVR também foi capaz de detectar rapidamente o desvio do valor da variável real em relação ao valor da variável calculada através do modelo de regressão.

Figura 4.32 - Rótulos obtidos através do processo de detecção do emp. F_f - SVR.



A Estatística T² de Hotelling foi utilizada para monitorar a modificação de operação normal para operação faltosa, como apresentado na Figura 4.33.

Figura 4.33 - Estatística T² - Emp. F_f - SVR.



A Tabela 4.4 apresenta os índices encontrados da Equação (3.1) para o sistema de detecção de falhas através de máquinas de vetores de suporte de regressão, o SVR.

Tabela 4.4 - Índices encontrados - SVR.

Oper.	Índices - SVR			
	DF/F	DF/N	DN/N	DN/F
Pert. C_{Af}	0,9875	0,25	0,75	0,0125
Falha C_B	0,99375	0,00	1,00	0,0063
Falha T	0,99375	0,00	1,00	0,0063
Emp. F_f	1,00	0,00	1,00	0,00

O método de detecção de falhas através do SVR não possui a capacidade de diagnosticar a operação faltosa, porém, foi capaz de detectar falhas com precisão e agilidade, detectando todas as operações faltosas em apenas um instante (0,05 h), apesar de não conseguir retomar a operação normal com agilidade como é possível verificar no caso da Perturbação C_{Af} devido à dinâmica das variáveis analisadas.

4.9.4. Discussão

A análise dos métodos de detecção e diagnóstico de falhas com utilização de máquinas de vetores de suporte foi realizada com o intuito de verificar as características positivas e negativas de cada método. Neste sentido, a Tabela 4.5 apresenta os valores dos índices apresentados na Seção 3.3, em que foram baseadas as conclusões acerca dos métodos SVM.

Tabela 4.5 - Índices - SVC OAO, SVC OAA e SVR.

Oper.	Índices	SVC OAO	SVC OAA	SVR
Pert. C_{Af}	DF/F	0,9813	0,9563	**0,9875**
	DF/N	0,0167	0,0667	0,2500
	DN/N	**0,9833**	0,8423	0,7500
	DN/F	0,0187	0,0437	0,0125
Falha C_B	DF/F	0,9708	0,9375	**0,9937**
	DF/N	0,0000	0,0000	0,0000
	DN/N	**1,0000**	**1,0000**	**1,0000**
	DN/F	0,0292	0,0625	0,0063
Falha T	DF/F	0,9833	**1,0000**	0,9937
	DF/N	0,0000	0,0000	0,0000
	DN/N	**1,0000**	**1,0000**	**1,0000**
	DN/F	0,0167	0,0000	0,0063
Emp. F_f	DF/F	0,9958	0,9958	**1,0000**
	DF/N	0,0000	0,0000	0,0000
	DN/N	**1,0000**	**1,0000**	**1,0000**
	DN/F	0,0042	0,0042	0,0000

Dentre os métodos de máquinas de vetores de suporte, pode-se destacar que o método de regressão (SVR) apresentou resultados consistentes, utilizando somente informações a respeito da operação normal, diferentemente dos métodos de classificação (SVC OAO e SVC OAA). A desvantagem do método SVR aparece na impossibilidade deste método identificar a falha, indicando apenas se há ou não uma anormalidade no sistema.

Dentre os métodos SVM estudados neste trabalho, pode-se verificar que para detectar operações anormais (índice DF/F), o método de regressão (SVR) foi o que apresentou um melhor desempenho para três casosdas situações avaliadas (perturbação C_{Af}, falha C_B e emperramento F_f). Para a falha T, o SVC OAO foi o método que obteve o melhor desempenho para detecção da falha, com DF/F = 0,9833. O método SVR também apresentou desempenho considerável na detecção de operação normal (índice DN/N), obtendo 100% de acerto na detecção para a falha C_B, a falha T e o emperramento F_f. No caso em que o sistema de detecção de falhas também identifica a falha, em metodologias com máquinas de vetores de classificação (SVC), a máquina de vetores de suporte de classificação *one-against-one* (SVC OAO) obteve melhores resultados na detecção da operação normal e na detecção das falhas. A máquina de vetores de suporte de classificação *one-against-all* (SVC OAA) obteve detecção perfeita (DF/F = 1,0000 e DN/N = 1,0000) para a falha T, detectando corretamente os instantes em que se encontrava a falha no processo.

4.9.5. Análise de Falhas com Diferentes Amplitudes

A aplicação de máquinas de vetores de classificação e de regressão para detectar falhas com diferentes amplitudes das falhas treinadas aos modelos anteriormente, para SVC OAO, SVC OAA e SVR, é estudada nos próximos tópicos.

Com o intuito de facilitar a comparação entre os resultados obtidos, os mesmos foram divididos de acordo com as falhas que foram propostas no Capítulo 3.

4.9.5.1. Perturbação C_{Af}

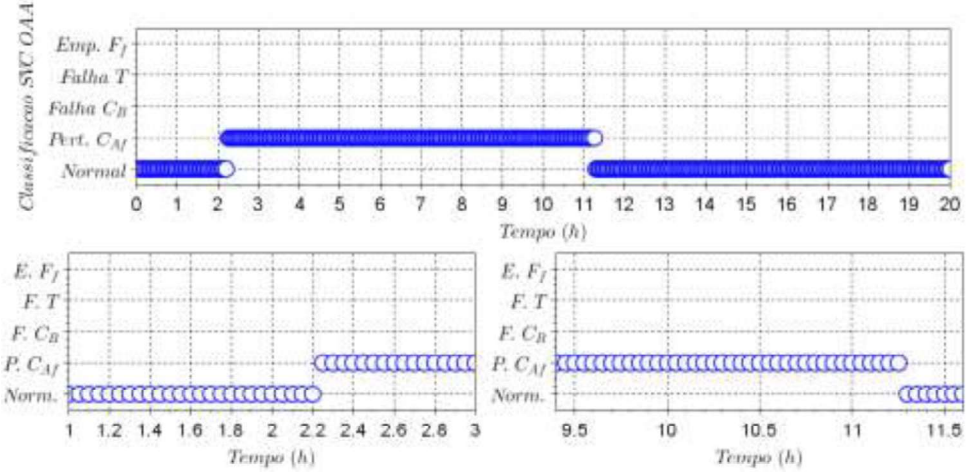
Na Seção 3.5.2, foram estabelecidas diferentes amplitudes das falhas pesquisadas e treinadas nos sistemas de detecção e diagnóstico de falhas. Para a perturbação C_{Af}, foram estabelecidos dois valores diferentes para C_{Af}, 4,5 mol/L e 4,9 mol/L.

A Figura 4.34 apresenta o resultado da detecção da perturbação C_{Af} pelo modelo de máquina de vetores de suporte de classificação *one-against-all* quando utilizado valor de C_{Af} menor (4,5 mol/L) que o valor de C_{Af} utilizado no treinamento (4,7 mol/L) para o método de detecção de falhas.

A perturbação C_{Af} menor (4,5 mol/L) foi detectada pelo modelo SVC OAA mais rapidamente que a perturbação C_{Af} de 4,7 mol/L treinada no modelo SVC OAA. Para os dados com amplitude diferente o sistema de detecção levou 4 intervalos de amostragem (0,20 h) para detectar a falha, enquanto que para os dados com amplitude igual aos dados de treinamento, o sistema de detecção levou 7 intervalos de amostragem (0,35 h). Após a perturbação C_{Af} ter sido removida, o sistema de detecção levou 9 intervalos de amostragem a mais (0,45 h) para detectar

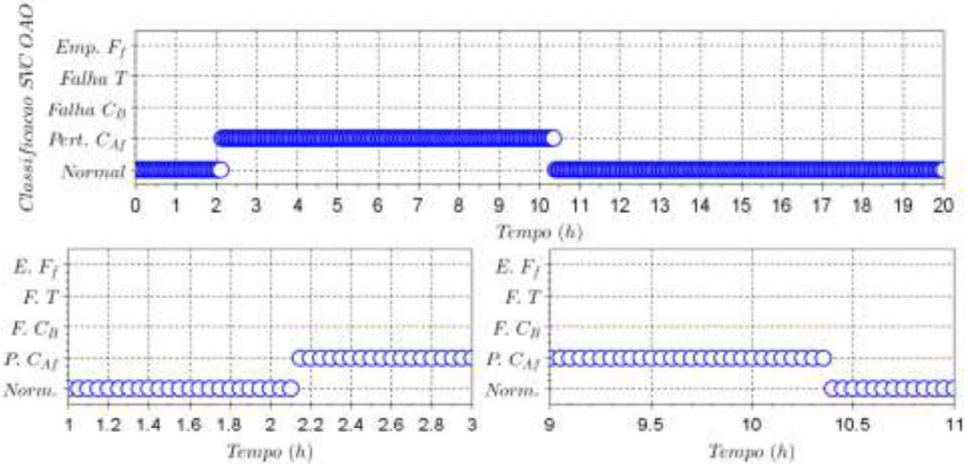
o retorno operação normal do que quando utilizados os dados com a mesma amplitude dos dados de treinamento (C_{Af} = 4,7 mol/L, 16 intervalos de amostragem, 0,8 h; C_{Af} = 4,5 mol/L, 25 intervalos de amostragem, 1,25 h).

Figura 4.34 - Rótulos obtidos através do processo de detecção da pert. C_{Af} (4,5 mol/L) - SVC OAA.



A Figura 4.35 apresenta o resultado da detecção da perturbação C_{Af} pelo modelo de máquina de vetores de suporte de classificação *one-against-one* quando utilizado valor de C_{Af} menor (4,5 mol/L) que o valor de C_{Af} utilizado no treinamento (4,7 mol/L) para o método de detecção de falhas.

Figura 4.35 - Rótulos obtidos através do processo de detecção da pert. C_{Af} (4,5 mol/L) - SVC OAO.

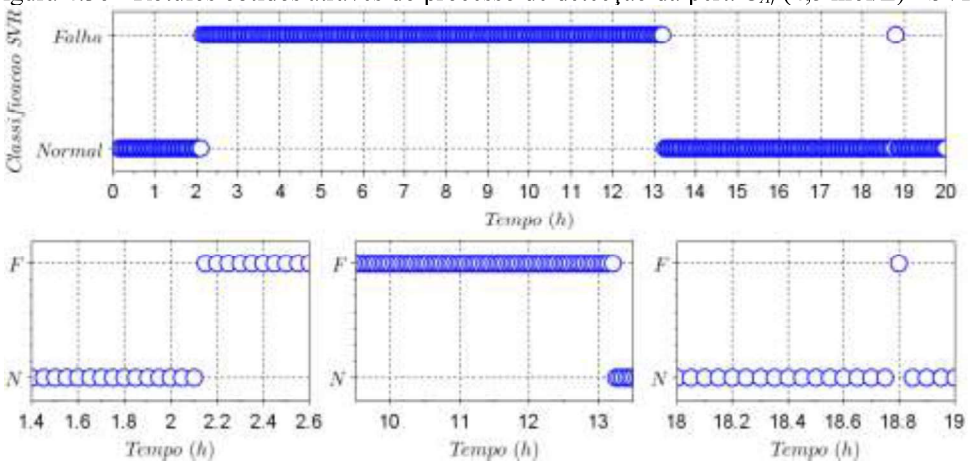


A perturbação C_{Af} menor (4,5 mol/L) foi detectada pelo modelo SVC OAO mais rapidamente que a perturbação C_{Af} de 4,7 mol/L treinada no modelo SVC OAO. Para os dados com amplitude diferente, o sistema de detecção levou 2 intervalos de amostragem (0,10 h) para detectar a falha, enquanto que para os dados com amplitude igual aos dados de treinamento, o sistema de detecção levou 3 intervalos de amostragem (0,15 h). Após a perturbação C_{Af} ter sido removida, o sistema de detecção levou 3 intervalos de amostragem a mais (0,15 h) para detectar

o retorno à operação normal do que quando utilizados os dados com a mesma amplitude dos dados de treinamento (C_{Af} = 4,7 mol/L, 4 intervalos de amostragem, 0,2 h; C_{Af} = 4,5 mol/L, 7 intervalos de amostragem, 0,35 h).

A Figura 4.36 apresenta o resultado da detecção da perturbação C_{Af}, pelo modelo de máquina de vetores de suporte de regressão quando utilizado valor de C_{Af} de 4,5 mol/L nos dados a serem detectados.

Figura 4.36 - Rótulos obtidos através do processo de detecção da pert. C_{Af} (4,5 mol/L) - SVR.



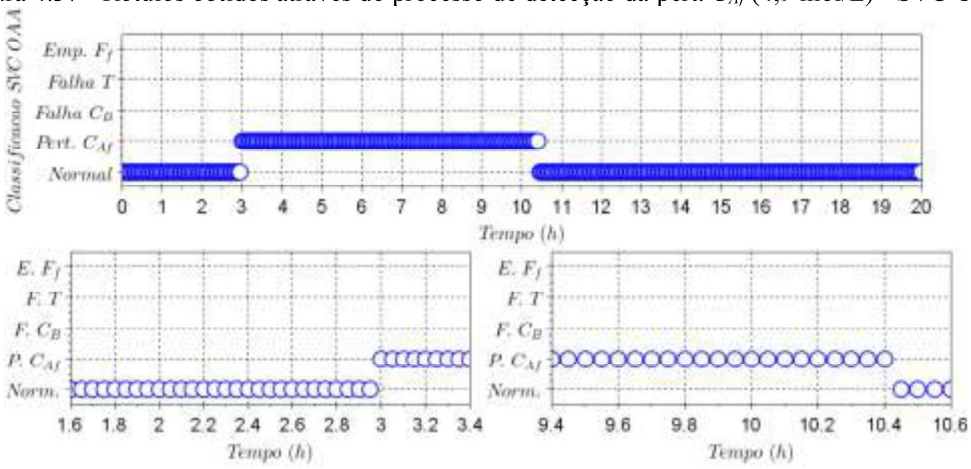
A perturbação C_{Af} menor (4,5 mol/L) foi detectada pelo modelo SVR dentro da mesma quantidade de intervalos de amostragem que a perturbação C_{Af} de 4,7 mol/L da mesma amplitude treinada no modelo SVR. Para ambos os casos, o sistema de detecção levou 2 intervalos de amostragem (0,10 h) para detectar falha. Após a perturbação C_{Af} ter sido removida, o sistema de detecção levou 4 intervalos de amostragem a mais (0,20 h) para detectar a retomada da operação normal do que o sistema que detectou a perturbação C_{Af} com valor de 4,7 mol/L (C_{Af} = 4,7 mol/L, 60 intervalos de amostragem, 3,0 h; C_{Af} = 4,5 mol/L, 64 intervalos de amostragem, 3,2 h). No intervalo de amostragem 18,8 h ocorreu a detecção de falha enquanto o sistema se encontrava em operação normal, da mesma forma que o SVR utilizado para detectar a perturbação C_{Af} igual a 4,7 mol/L.

A Figura 4.37 apresenta o resultado da detecção da perturbação C_{Af} pelo modelo de máquina de vetores de suporte de classificação *one-against-all* quando utilizado valor de C_{Af} maior (4,9 mol/L) que o valor de C_{Af} utilizado no treinamento (4,7 mol/L) do método de detecção de falhas.

A perturbação C_{Af} maior (4,9 mol/L) foi detectada pelo modelo SVC OAA mais lentamente que a perturbação C_{Af} de 4,7 mol/L treinada no modelo SVC OAA. Para os dados com amplitude diferente, o sistema de detecção levou 19 intervalos de amostragem (0,95 h) para detectar a falha, enquanto que para os dados com amplitude igual aos dados de treinamento

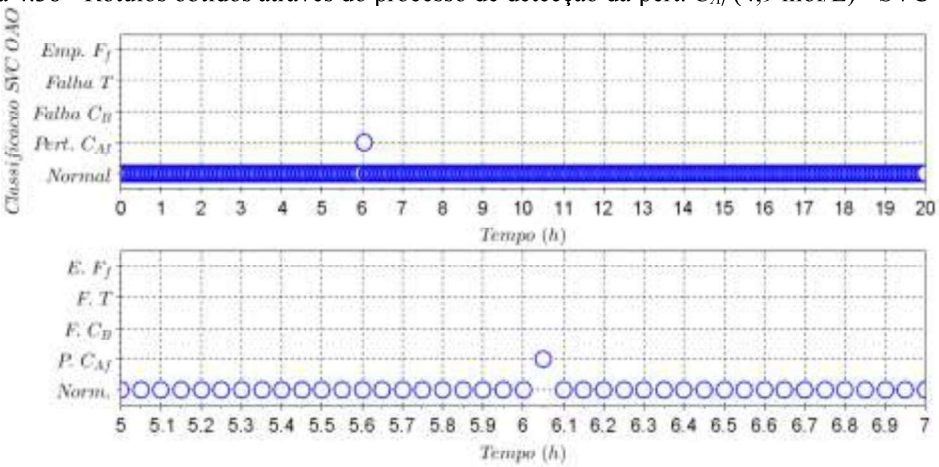
o sistema de detecção levou 7 intervalos de amostragem (0,35 h). Após a perturbação C_{Af} ter sido removida, o sistema de detecção levou 8 intervalos de amostragem a menos (0,4 h) para detectar o retorno à operação normal do que quando utilizados os dados com a mesma amplitude dos dados de treinamento (C_{Af} = 4,7 mol/L, 16 intervalos de amostragem, 0,8 h; C_{Af} = 4,9 mol/L, 8 intervalos de amostragem, 0,4 h).

Figura 4.37 - Rótulos obtidos através do processo de detecção da pert. C_{Af} (4,9 mol/L) - SVC OAA.



A Figura 4.38 apresenta o resultado da detecção da perturbação C_{Af} pelo modelo de máquina de vetores de suporte de classificação *one-against-one* quando utilizado valor de C_{Af} maior (4,9 mol/L) que o valor de C_{Af} utilizado no treinamento (4,7 mol/L) do método de detecção de falhas.

Figura 4.38 - Rótulos obtidos através do processo de detecção da pert. C_{Af} (4,9 mol/L) - SVC OAO.

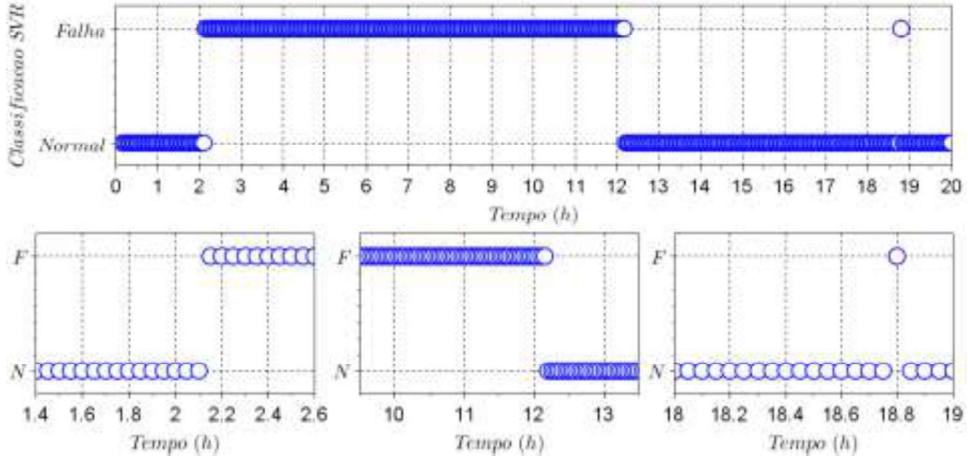


A perturbação C_{Af} maior (4,9 mol/L) não foi detectada pelo modelo SVC OAO. Para os dados com amplitude diferente, o sistema de detecção detectou a perturbação C_{Af} somente no instante 6,05 h. Isto ocorre devido as modificações das variáveis de entrada não serem suficientes para que o sistema de detecção de falhas, treinado com os dados originais (C_{Af} = 4,7

mol/L), fosse capaz de classificar os dados simulados com C_{Af} igual a 4,9 mol/L.

A Figura 4.39 apresenta o resultado da detecção da perturbação C_{Af} pelo modelo de máquina de vetores de suporte de regressão quando utilizado valor de C_{Af} de 4,9 mol/L.

Figura 4.39 - Rótulos obtidos através do processo de detecção da pert. C_{Af} (4,9 mol/L) - SVR.



A perturbação C_{Af} maior (4,9 mol/L) foi detectada pelo modelo SVR dentro da mesma quantidade de intervalos de amostragem que a perturbação C_{Af} de 4,7 mol/L foi detectada pelo modelo SVR. Para ambos os casos, o sistema de detecção levou 2 intervalos de amostragem (0,10 h) após a aplicação da perturbação para detectar a operação faltosa. Após a perturbação C_{Af} ter sido removida, o sistema de detecção levou 4 intervalos de amostragem a mais (0,20 h) para detectar o retorno a operação normal do que o sistema que detectou a perturbação C_{Af} de com valor de 4,7 mol/L (C_{Af} = 4,7 mol/L, 60 intervalos de amostragem, 3,0 h; C_{Af} = 4,5 mol/L, 64 intervalos de amostragem, 3,2 h). Existiu um intervalo de amostragem no instante 18,8 h em que houve a detecção de falha enquanto o sistema se encontrava em operação normal, da mesma forma que o SVR utilizado para detectar a perturbação C_{Af} igual a 4,7 mol/L.

A Tabela 4.6 apresenta os instantes de detecção para as diferentes metodologias e diferentes amplitudes da perturbação C_{Af}, assim como os índices obtidos para cada situação.

Tabela 4.6 - Índices - SVC OAO, SVC OAA e SVR - amplitudes perturbação C_{Af}.

C_{Af}	Metodologia	TAD (h)	DF/F	DF/N	DN/N	DN/F
4,5 mol/L	SVC OAA	2,25	0,9750	0,1042	0,8958	0,0250
	SVC OAO	**2,15**	**0,9875**	0,0292	**0,9708**	0,0125
	SVR	**2,15**	**0,9875**	0,2708	0,7292	0,0125
4,7 mol/L	SVC OAA	2,40	0,9563	0,0667	**0,9333**	0,0437
	SVC OAO	2,20	0,9813	0,0167	**0,9833**	0,0187
	SVR	**2,15**	**0,9875**	0,2542	0,7458	0,0125
4,9 mol/L	SVC OAA	3,00	0,8813	0,0333	**0,9667**	0,1187
	SVC OAO	-	-	-	-	-
	SVR	**2,15**	**0,9875**	0,1792	0,8208	0,0125

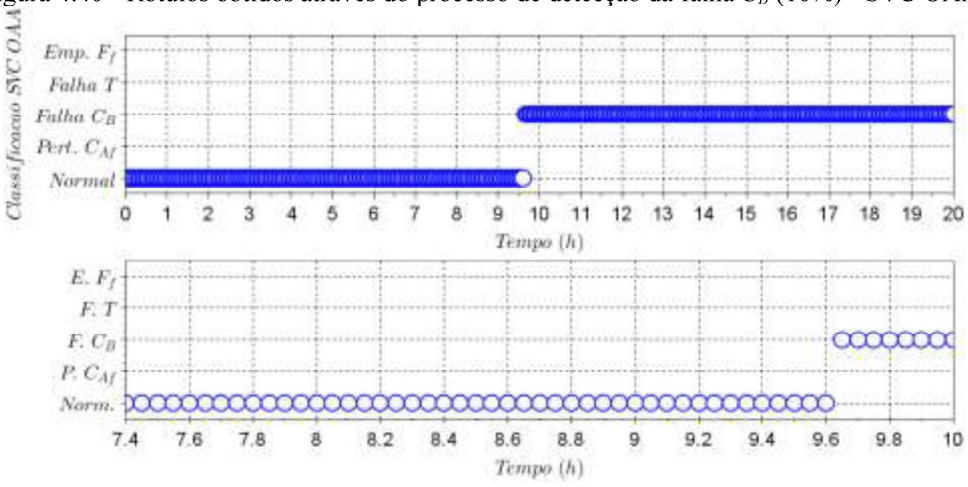
Para as diferentes amplitudes da perturbação C_{Af} os modelos que reconheceram valores maiores e valores menores de C_{Af} foram as metodologias SVM de classificação *one-against-all* e SVM de regressão, apesar do último não classificar a falha detectada. A metodologia SVM de classificação *one-against-one* não conseguiu detectar a perturbação C_{Af} com o valor de C_{Af} igual a 4,9 mol/L. Neste caso, como a variação de C_{Af} (5,1 – 4,9 = 0,2 mol/L) foi muito pequena, o método não foi capaz de diferenciar da operação normal os dados com a perturbação C_{Af}. Entre as diferenças dos métodos, o SVR foi o único capaz de identificar de maneira consistente em todas as três situações com diferentes C_{Af}, levando três intervalos de amostragem para detectar a falha.

4.9.5.2. Falha C_B

Na Seção 3.5.2, foram estabelecidas diferentes amplitudes das falhas pesquisadas e treinadas nos sistemas de detecção e diagnóstico de falhas. Para a falha C_B, foram estabelecidos aumentos abruptos menor (10%) e maior (30%) no valor medido da variável controlada C_B após a aplicação da falha, diferentes do aumento abrupto utilizado para treinamento (20%).

A Figura 4.40 apresenta o resultado da detecção da falha C_B pelo modelo de máquina de vetores de suporte de classificação *one-against-all* quando utilizado aumento abrupto menor (10% do valor medido de C_B) que o aumento abrupto utilizado no treinamento (20%) do método de detecção de falhas.

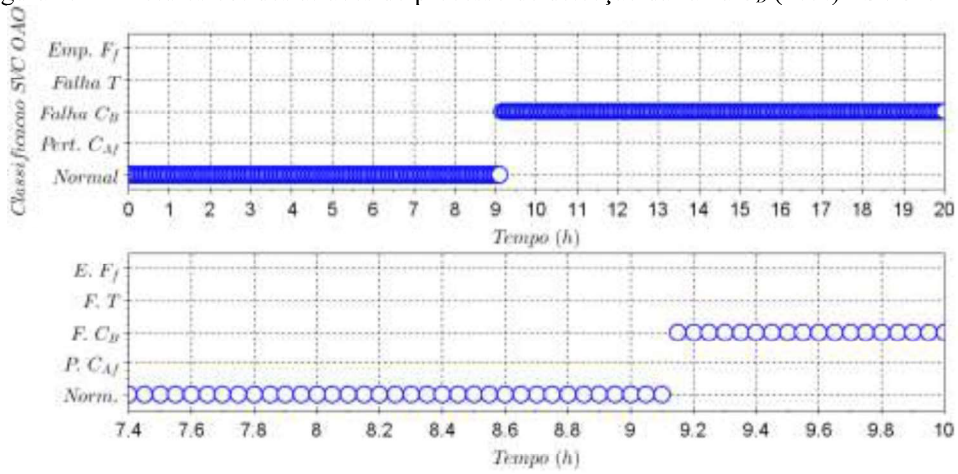
Figura 4.40 - Rótulos obtidos através do processo de detecção da falha C_B (10%) - SVC OAA.



A falha C_B obtida através do aumento abrupto menor (10%) foi detectada pelo modelo SVC OAA mais lentamente que a falha C_B obtida com o aumento abrupto de 20% da variável medida. Para os dados com amplitude diferente, o sistema de detecção levou 32 intervalos de amostragem, ou 1,60 h, para detectar a falha, enquanto que para os dados com amplitude igual aos dados de treinamento, o sistema de detecção levou 15 intervalos de amostragem, ou 0,75 h.

A Figura 4.41 apresenta o resultado da detecção da falha C_B pelo modelo de máquina de vetores de suporte de classificação *one-against-one* quando utilizado aumento abrupto menor (10% do valor medido de C_B) que o aumento abrupto utilizado no treinamento (20%) do método de detecção de falhas.

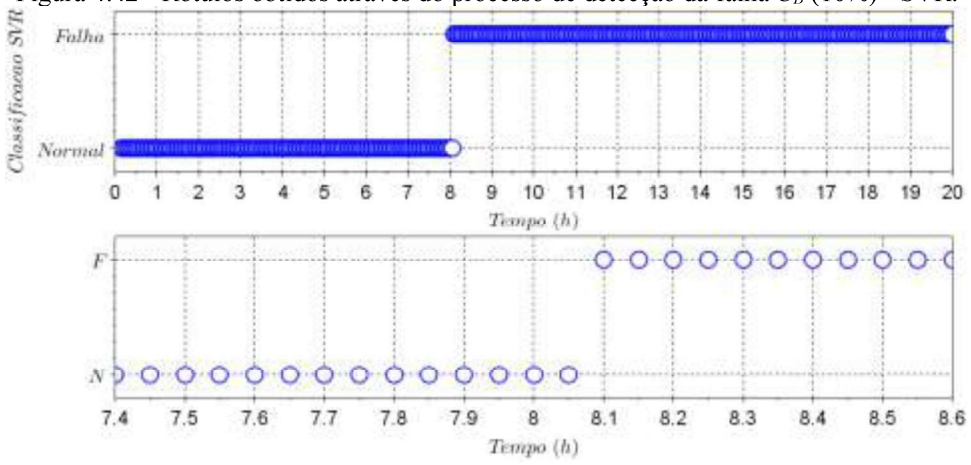
Figura 4.41 - Rótulos obtidos através do processo de detecção da falha C_B (10%) - SVC OAO.



A falha C_B obtida através do aumento abrupto menor (10%) foi detectada pelo modelo SVC OAO mais lentamente que a falha C_B obtida com o aumento abrupto de 20% da variável medida. Para os dados com amplitude diferente, o sistema de detecção levou 22 intervalos de amostragem (1,10 h) para detectar a falha, enquanto que para os dados com amplitude igual aos dados de treinamento, o sistema de detecção levou 7 intervalos de amostragem (0,35 h).

A Figura 4.42 apresenta o resultado da detecção da falha C_B pelo modelo de máquina de vetores de suporte de regressão quando utilizado aumento abrupto menor (10% do valor medido de C_B) que o aumento abrupto apresentado na Seção 4.9.3 (20% do valor medido de C_B).

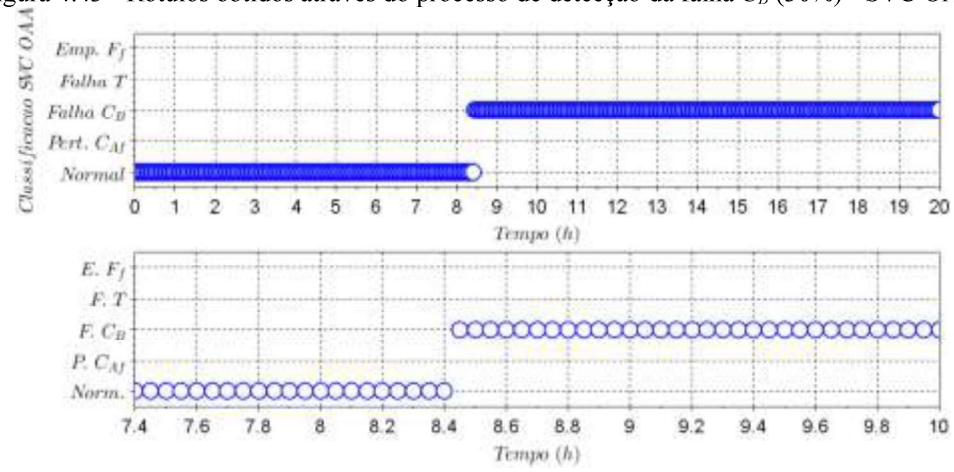
Figura 4.42 - Rótulos obtidos através do processo de detecção da falha C_B (10%) - SVR.



A falha C_B obtida através do aumento abrupto de 10% da variável C_B foi detectada pelo modelo SVR dentro da mesma quantidade de intervalos de amostragem que os dados com aumento abrupto de 20% da variável C_B. Para ambos os casos, o sistema de detecção levou 1 intervalo de amostragem (0,05 h) para detectar falha.

A Figura 4.43 apresenta o resultado da detecção da falha C_B utilizando modelo de máquina de vetores de suporte de classificação *one-against-all* obtida através dos dados com aumento abrupto de 30% da variável controlada C_B.

Figura 4.43 - Rótulos obtidos através do processo de detecção da falha C_B (30%) - SVC OAA.

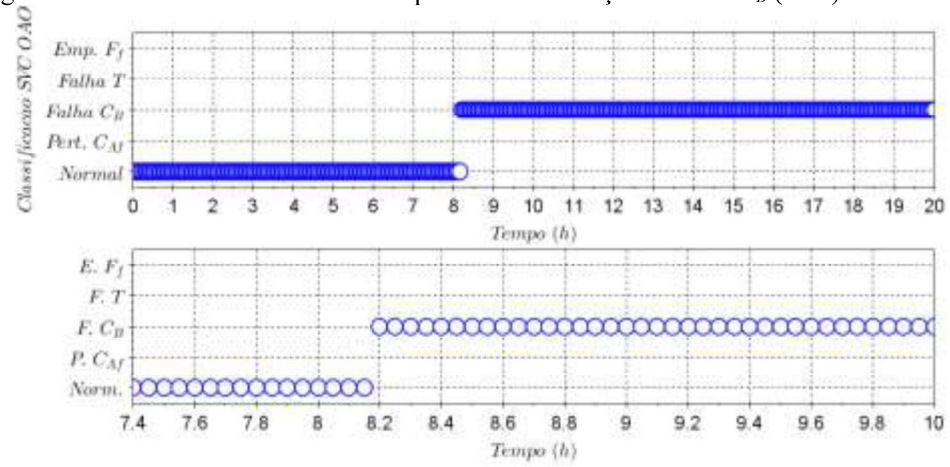


A falha C_B obtida através do maior aumento abrupto da variável controlada (30%) foi detectada pelo modelo SVC OAA mais rapidamente que a falha C_B obtida através do aumento abrupto de 20%. Para os dados com amplitude diferente, o sistema de detecção levou 8 intervalos de amostragem (0,40 h) para detectar a falha, enquanto que para os dados com amplitude igual aos dados de treinamento, o sistema de detecção levou 15 intervalos de amostragem (0,75 h).

A Figura 4.44 apresenta o resultado da detecção da falha C_B utilizando modelo de máquina de vetores de suporte de classificação *one-against-one* obtida através dos dados com aumento abrupto de 30% da variável controlada C_B.

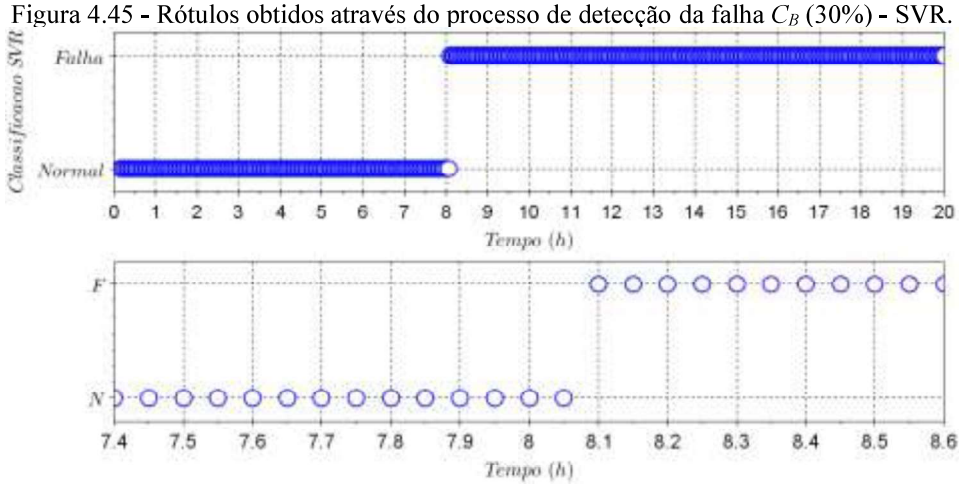
A falha C_B obtida através do maior aumento abrupto da variável controlada (30%) foi detectada pelo modelo SVC OAO mais rapidamente que a falha C_B obtida através do aumento abrupto de 20%. Para os dados com amplitude diferente, o sistema de detecção levou 3 intervalos de amostragem (0,15 h) para detectar a falha, enquanto que para os dados com amplitude igual aos dados de treinamento, o sistema de detecção levou 7 intervalos de amostragem (0,35 h).

Figura 4.44 - Rótulos obtidos através do processo de detecção da falha C_B (30%) - SVC OAO.



A Figura 4.45 apresenta o resultado da detecção da falha C_B pelo modelo de máquina de vetores de suporte de regressão quando utilizado aumento abrupto maior (30% do valor medido de C_B) que o aumento abrupto apresentado na Seção 4.9.3 (20% do valor medido de C_B).

Figura 4.45 - Rótulos obtidos através do processo de detecção da falha C_B (30%) - SVR.



A falha C_B obtida através do aumento abrupto de 30% da variável C_B foi detectada pelo modelo SVR dentro da mesma quantidade de intervalos de amostragem que os dados com aumento abrupto de 20% da variável C_B. Para ambos os casos, o sistema de detecção levou 1 intervalo de amostragem (0,05 h) para detectar falha.

A Tabela 4.7 apresenta os instantes de detecção para as diferentes metodologias e diferentes amplitudes da falha C_B, assim como os índices obtidos para cada situação.

Todos os modelos reconheceram as diferentes amplitudes da falha C_B. Entre as metodologias SVM de classificação, a SVC *one-against-one* detectou a falha C_B mais rapidamente que a metodologia SVC *one-against-all*. Entre as diferentes metodologias, a SVR foi a única capaz de identificar de maneira consistente em todas as três situações com diferentes

C_B levando um intervalo de amostragem para detectar a falha, além de ser a metodologia que mais rapidamente detecta a falha, apesar de não a identificar.

Tabela 4.7 - Índices - SVC OAA, SVC OAO e SVR - amplitudes falha C_B.

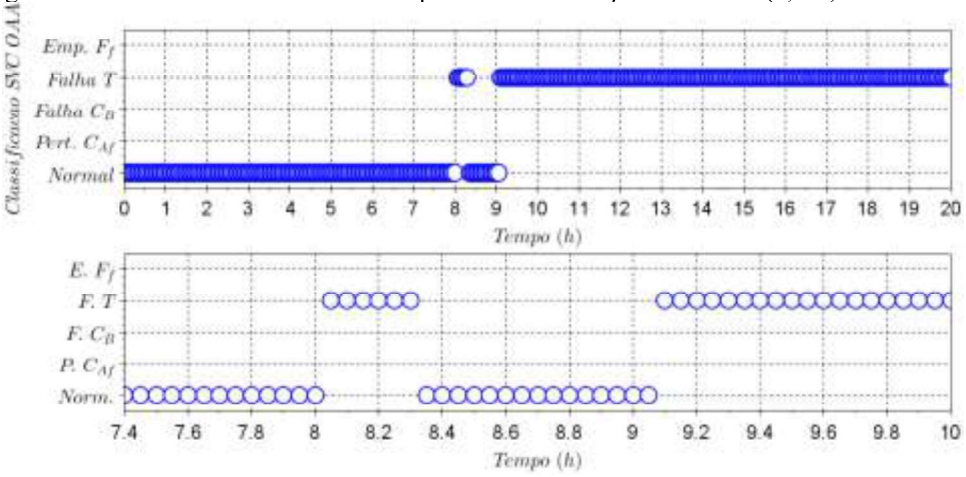
C_B	Metodologia	TAD (h)	DF/F	DF/N	DN/N	DN/F
10%	SVC OAA	9,65	0,8667	0,0000	1,0000	0,1333
	SVC OAO	9,15	0,9083	0,0000	1,0000	0,0917
	SVR	**8,10**	**0,9958**	0,0000	1,0000	0,0042
20%	SVC OAA	8,80	0,9375	0,0000	1,0000	0,0625
	SVC OAO	8,40	0,9708	0,0000	1,0000	0,0292
	SVR	**8,10**	**0,9958**	0,0000	1,0000	0,0042
30%	SVC OAA	8,45	0,9667	0,0000	1,0000	0,0333
	SVC OAO	8,20	0,9875	0,0000	1,0000	0,0125
	SVR	**8,10**	**0,9958**	0,0000	1,0000	0,0042

4.9.5.3. Falha *T*

Na Seção 3.5.2, foram estabelecidas diferentes amplitudes das falhas pesquisadas e treinadas nos sistemas de detecção e diagnóstico de falhas. Para a falha *T*, foram estabelecidas medidas incorretas do sensor de temperatura do reator (0,5% e 2,0% maiores que a medida imediatamente anterior a aplicação da falha) da variável controlada *T*, diferentes da medida incorreta do sensor de temperatura do reator utilizada para treinamento (1,0% maior).

A Figura 4.46 apresenta o resultado da detecção da falha *T* pelo modelo de máquina de vetores de suporte de classificação *one-against-all* quando utilizada medida incorreta do sensor de temperatura menor (0,5% maior que a medida de *T*) que a medida incorreta utilizada no treinamento (1,0% maior) do método de detecção de falhas.

Figura 4.46 - Rótulos obtidos através do processo de detecção da falha *T* (0,5%) - SVC OAA.

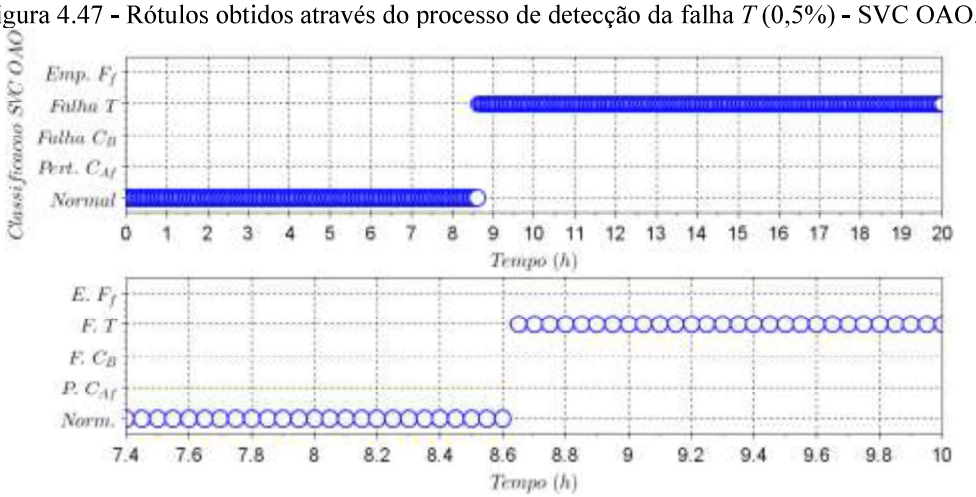


A falha *T* obtida através da medida incorreta do sensor *T* menor (0,5%) foi detectada pelo modelo SVC OAA instantaneamente, porém devido a dinâmica dos sinais manipulados e controlados, após 6 intervalos de amostragem (0,30 h) o sistema de detecção voltou a detectar

a operação normal durante 15 intervalos de amostragem (0,75 h) até o instante 9,1 h. Após os 15 intervalos de amostragem, o sistema voltou a detectar a falha *T*.

A Figura 4.47 apresenta o resultado da detecção da falha *T* pelo modelo de máquina de vetores de suporte de classificação *one-against-one* quando utilizada medida incorreta do sensor de temperatura menor (0,5% maior que a medida de *T*) que a medida incorreta utilizada no treinamento (1,0% maior) do método de detecção de falhas.

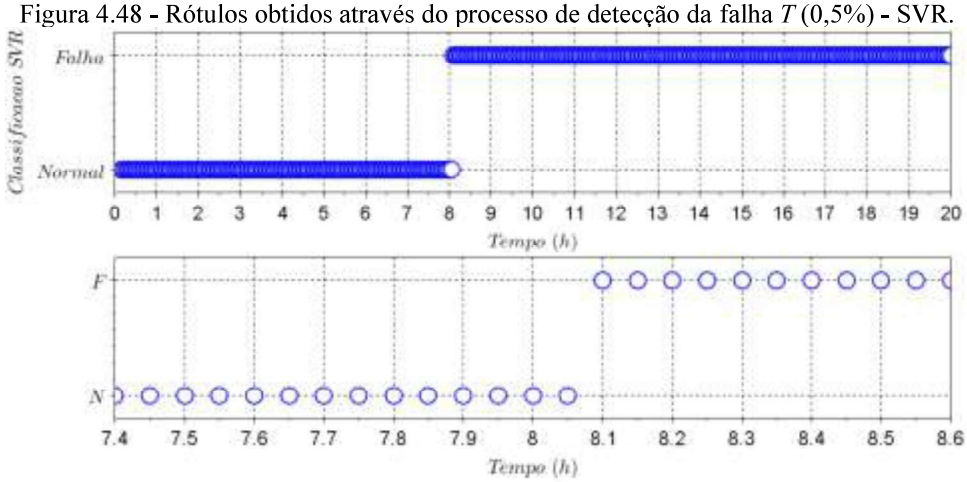
Figura 4.47 - Rótulos obtidos através do processo de detecção da falha *T* (0,5%) - SVC OAO.



A falha *T* obtida através da medida incorreta do sensor *T* menor (0,5% maior que a medida incorreta anterior a aplicação da falha) foi detectada pelo modelo SVC OAO após 12 intervalos de amostragem (0,60 h), mais lentamente que a medida incorreta do sensor *T* utilizada no treinamento (1,0% maior), detectada após 4 intervalos de amostragem (0,20 h).

A Figura 4.48 apresenta o resultado da detecção da falha *T* pelo modelo de máquina de vetores de suporte de regressão quando a medida incorreta é menor (0,5% maior que a medida incorreta do sensor de temperatura *T*) que a medida incorreta apresentada na Seção 4.9.3 (1,0% maior).

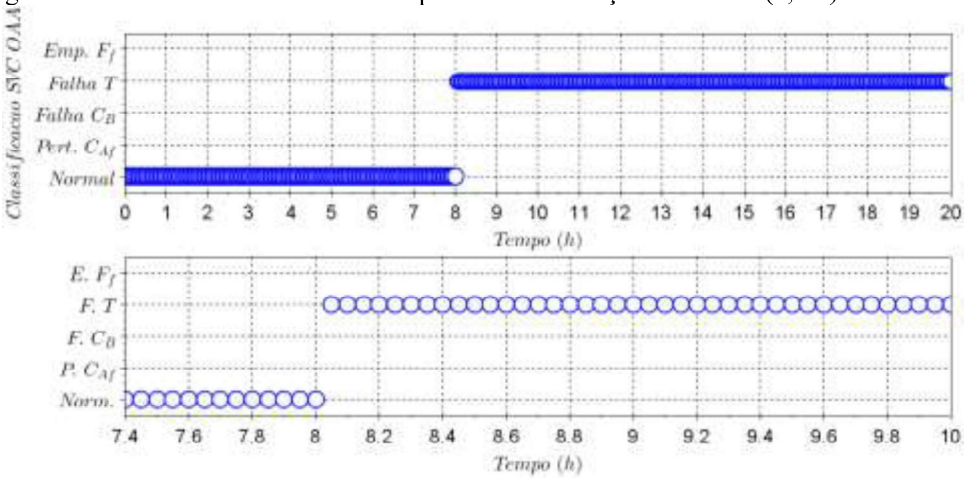
Figura 4.48 - Rótulos obtidos através do processo de detecção da falha *T* (0,5%) - SVR.



A falha *T* obtida através da medida incorreta do sensor de temperatura *T* do reator 0,5% maior que a medida anterior a aplicação da falha *T* foi detectada pelo modelo SVR dentro da mesma quantidade de intervalos de amostragem que os dados com medida incorreta 1,0% maior da variável *T*. Para ambos os casos, o sistema de detecção levou 1 intervalo de amostragem (0,05 h) para detectar falha.

A Figura 4.49 apresenta o resultado da detecção da falha *T* pelo modelo de máquina de vetores de suporte de classificação *one-against-all* quando utilizada medida incorreta do sensor de temperatura maior (2,0% maior que a medida de *T*) que a medida incorreta utilizada no treinamento (1,0% maior) do método de detecção de falhas.

Figura 4.49 - Rótulos obtidos através do processo de detecção da falha *T* (2,0%) - SVC OAA.

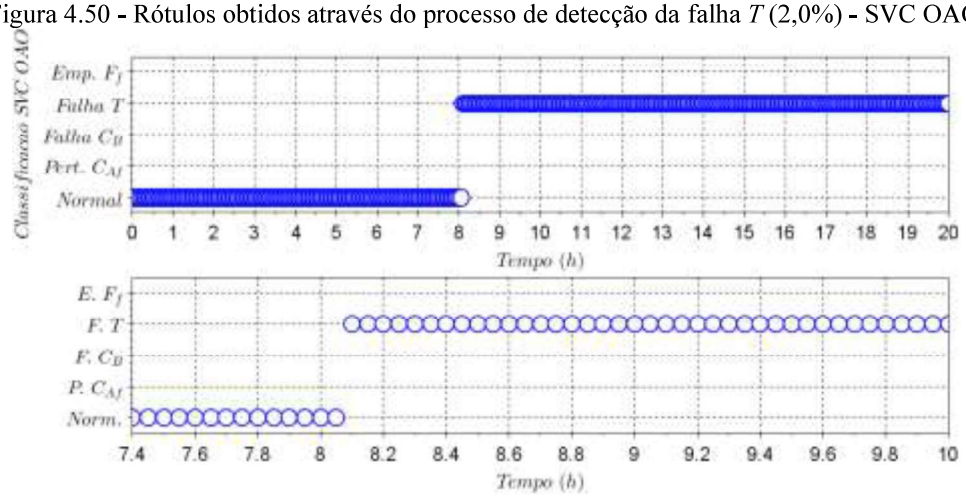


A falha *T* obtida através da medida incorreta do sensor *T* maior (2,0%) foi detectada pelo modelo SVC OAA instantaneamente, de maneira idêntica a forma que o modelo detectou a falha *T* com dados obtidos através da medida incorreta do sensor *T* 1,0% maior que a última medida correta.

A Figura 4.50 apresenta o resultado da detecção da falha *T* pelo modelo de máquina de vetores de suporte de classificação *one-against-one* quando utilizada medida incorreta do sensor de temperatura maior (2,0% maior que a medida de *T*) que a medida incorreta utilizada no treinamento (1,0% maior) do método de detecção de falhas.

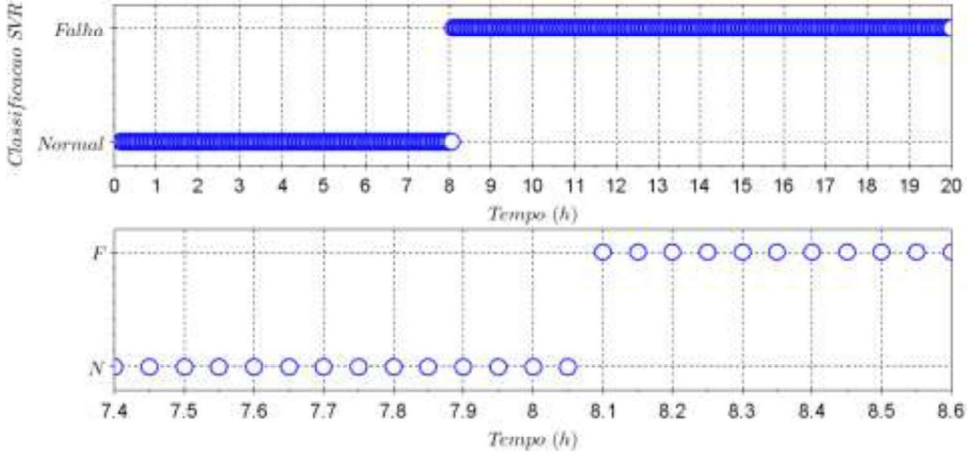
A falha *T* obtida através da medida incorreta do sensor *T* maior (2,0% maior que a medida incorreta anterior a aplicação da falha) foi detectada pelo modelo SVC OAO após 1 intervalo de amostragem (0,05 h), mais rapidamente que a medida incorreta do sensor *T* utilizada no treinamento (1,0% maior), detectada após 4 intervalos de amostragem (0,20 h).

Figura 4.50 - Rótulos obtidos através do processo de detecção da falha *T* (2,0%) - SVC OAO.



A Figura 4.51 apresenta o resultado da detecção da falha *T* pelo modelo de máquina de vetores de suporte de regressão quando a medida incorreta é maior (2,0% maior que a medida incorreta do sensor de temperatura *T*) que a medida incorreta apresentada na Seção 4.9.3 (1,0% maior).

Figura 4.51 - Rótulos obtidos através do processo de detecção da falha *T* (2,0%) - SVR.



A falha *T* obtida através da medida incorreta do sensor de temperatura *T* do reator 2,0% maior que a medida anterior a aplicação da falha *T* foi detectada pelo modelo SVR dentro da mesma quantidade de intervalos de amostragem que os dados com medida incorreta 1,0% maior da variável *T*. Para ambos os casos, o sistema de detecção levou 1 intervalo de amostragem (0,05 h) para detectar falha.

A Tabela 4.8 apresenta os instantes de detecção para as diferentes metodologias e diferentes amplitudes da falha *T*, assim como os índices obtidos para cada situação.

Todos os modelos reconheceram as diferentes amplitudes da falha *T*. Entre as metodologias SVM de classificação, a SVC *one-against-all* detectou a falha *T* mais rapidamente que a metodologia SVC *one-against-one*, porém o modelo SVC OAA não

conseguiu detectar a falha *T* com medida incorreta 0,5% maior que a última medida válida durante todos os instantes em que a falha *T* foi aplicada. Entre as diferentes metodologias, a SVR foi a única capaz de identificar de maneira consistente em todas as três situações com diferentes *T* detectando a falha após um intervalo de amostragem, além de ser a metodologia que mais rapidamente detecta a falha, apesar de não a identificar.

Tabela 4.8 - Índices - SVC OAA, SVC OAO e SVR - amplitudes falha *T*.

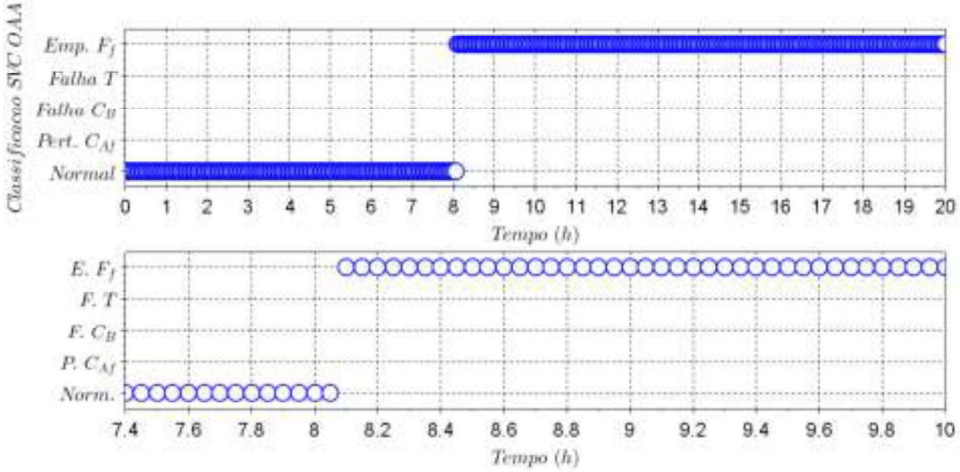
T	Metodologia	TAD (h)	DF/F	DF/N	DN/N	DN/F
0,5%	SVC OAA	**8,05**	0,9063	0,0000	1,0000	0,0937
	SVC OAO	8,65	0,9500	0,0000	1,0000	0,0500
	SVR	8,10	**0,9917**	0,0000	1,0000	0,0083
1%	SVC OAA	**8,05**	**1,0000**	0,0000	1,0000	0,0000
	SVC OAO	8,25	0,9833	0,0000	1,0000	0,0167
	SVR	8,10	0,9917	0,0000	1,0000	0,0083
2%	SVC OAA	**8,05**	**1,0000**	0,0000	1,0000	0,0000
	SVC OAO	8,10	0,9917	0,0000	1,0000	0,0083
	SVR	8,10	0,9917	0,0000	1,0000	0,0083

4.9.5.4. Emperramento F_f

Na Seção 3.5.2, foram estabelecidas diferentes amplitudes das falhas pesquisadas e treinadas nos sistemas de detecção e diagnóstico de falhas. Para o emperramento F_f, foram estabelecidos dados cuja vazão foi estabelecida a partir do momento em que foi aplicada a falha em 60% (menor) e 80% (maior) da vazão de alimentação do reator em estado estacionário, diferentes da vazão utilizada para os dados do treinamento dos métodos do emperramento F_f (70% da vazão de alimentação do reator em estado estacionário).

A Figura 4.52 apresenta o resultado da detecção do emperramento F_f pelo modelo de máquina de vetores de suporte de classificação *one-against-all* quando utilizada 60% da vazão de estado estacionário, vazão menor que a vazão de alimentação do reator utilizada no treinamento (70% da vazão de estado estacionário) do método de detecção de falhas.

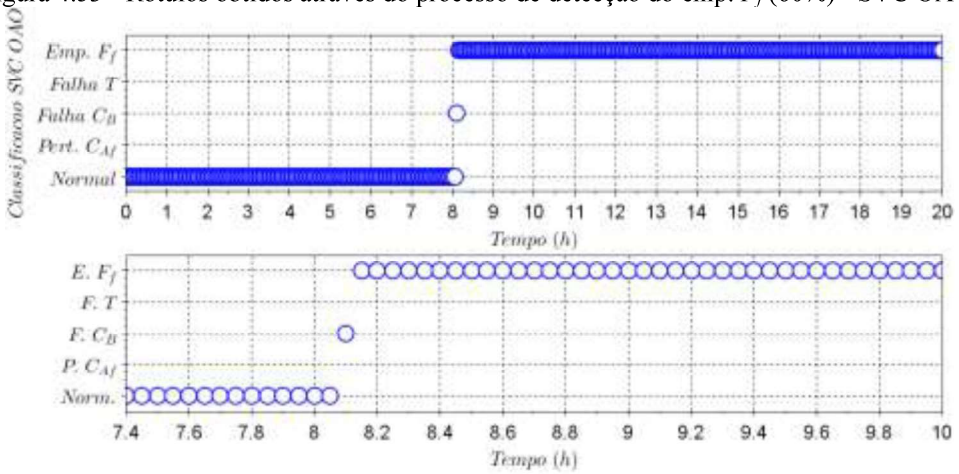
Figura 4.52 - Rótulos obtidos através do processo de detecção do emp. F_f (60%) - SVC OAA.



O emperramento F_f obtido através da modificação da vazão de alimentação do reator devido a um emperramento da válvula, em que a vazão passa a ser 60% da vazão de estado estacionário, foi detectada pelo modelo SVC OAA após um intervalo de amostragem (0,05 h).

A Figura 4.53 apresenta o resultado da detecção do emperramento F_f pelo modelo de máquina de vetores de suporte de classificação *one-against-one* quando utilizada 60% da vazão de alimentação do reator no estado estacionário, vazão menor que vazão utilizada no treinamento (70% da vazão no estado estacionário) do método de detecção de falhas.

Figura 4.53 - Rótulos obtidos através do processo de detecção do emp. F_f (60%) - SVC OAO.

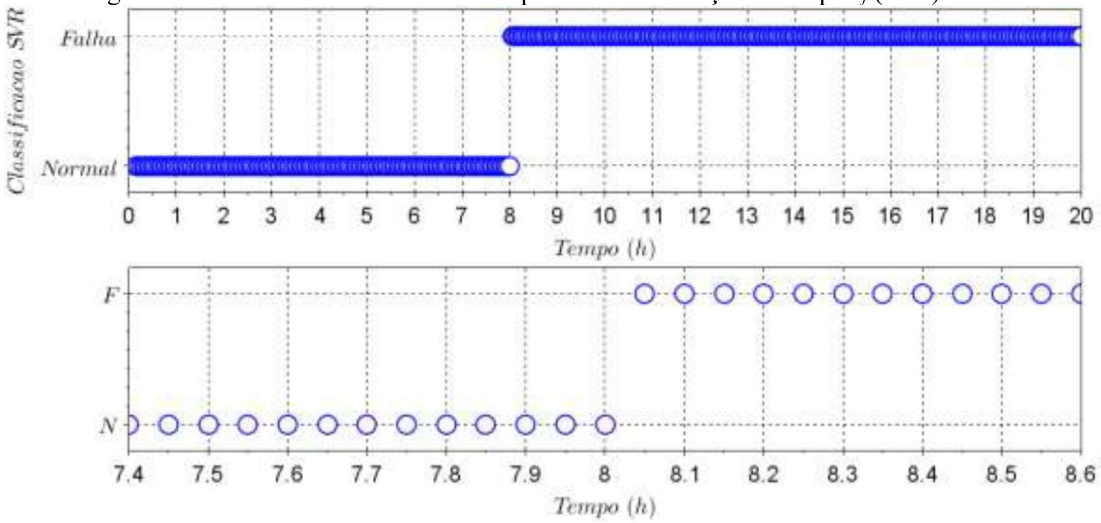


O emperramento F_f obtido através da modificação da vazão de alimentação do reator devido a um emperramento da válvula, em que a vazão passa a ser 60% da vazão de estado estacionário, foi detectada pelo modelo SVC OAO após um intervalo de amostragem (0,05 h), de forma idêntica que o sistema detectou o emperramento F_f utilizado no treinamento (70% da vazão de alimentação do reator no estado estacionário). Porém, quando utilizados os dados com emperramento da válvula em 60% da vazão no estado estacionário, o sistema de detecção comete um erro durante o primeiro intervalo de detecção, no instante 8,10 h, e detecta a falha C_B ao invés do emperramento F_f. A partir do instante 8,15 h, o sistema passa a detectar o emperramento F_f corretamente.

A Figura 4.54 apresenta o resultado da detecção do emperramento F_f pelo modelo de máquina de vetores de suporte de regressão quando a vazão de alimentação é menor (60% da vazão de alimentação do reator no estado estacionário) que a vazão de alimentação apresentada na Seção 4.9.3 (70% da vazão no estado estacionário).

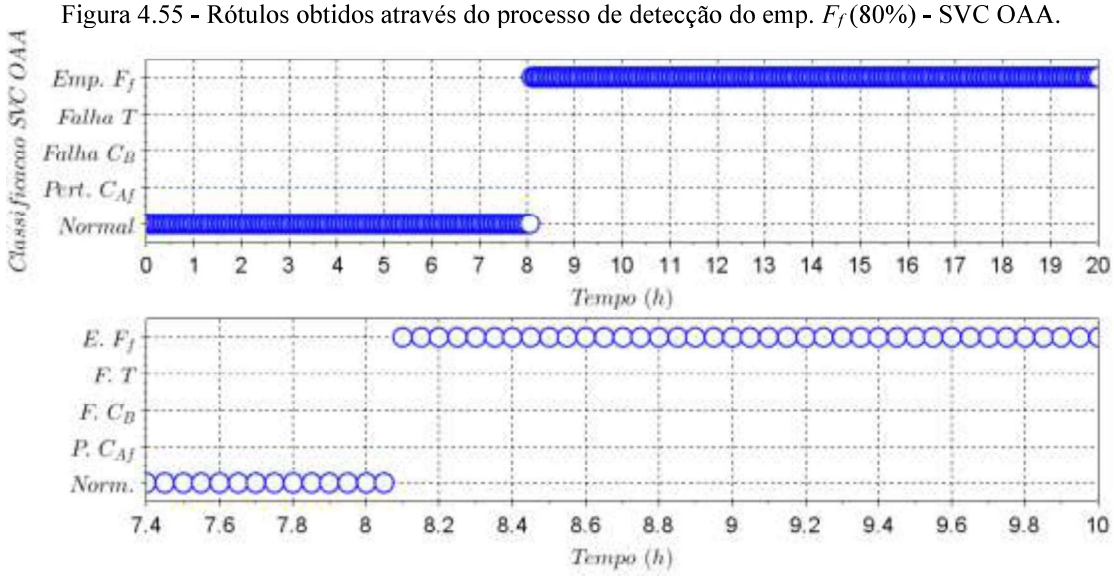
O emperramento F_f obtido através de 60% da vazão de alimentação do reator no estado estacionário foi detectado pelo modelo SVR dentro da mesma quantidade de intervalos de amostragem que os dados com 70% da vazão em estado estacionário. Para ambos os casos, a detecção ocorreu instantaneamente.

Figura 4.54 - Rótulos obtidos através do processo de detecção do emp. F_f (60%) - SVR.



A Figura 4.55 apresenta o resultado da detecção do emperramento F_f pelo modelo de máquina de vetores de suporte de classificação *one-against-all* quando utilizada 80% da vazão de estado estacionário, vazão maior que a vazão de alimentação do reator utilizada no treinamento (70% da vazão de estado estacionário) do método de detecção de falhas.

Figura 4.55 - Rótulos obtidos através do processo de detecção do emp. F_f (80%) - SVC OAA.

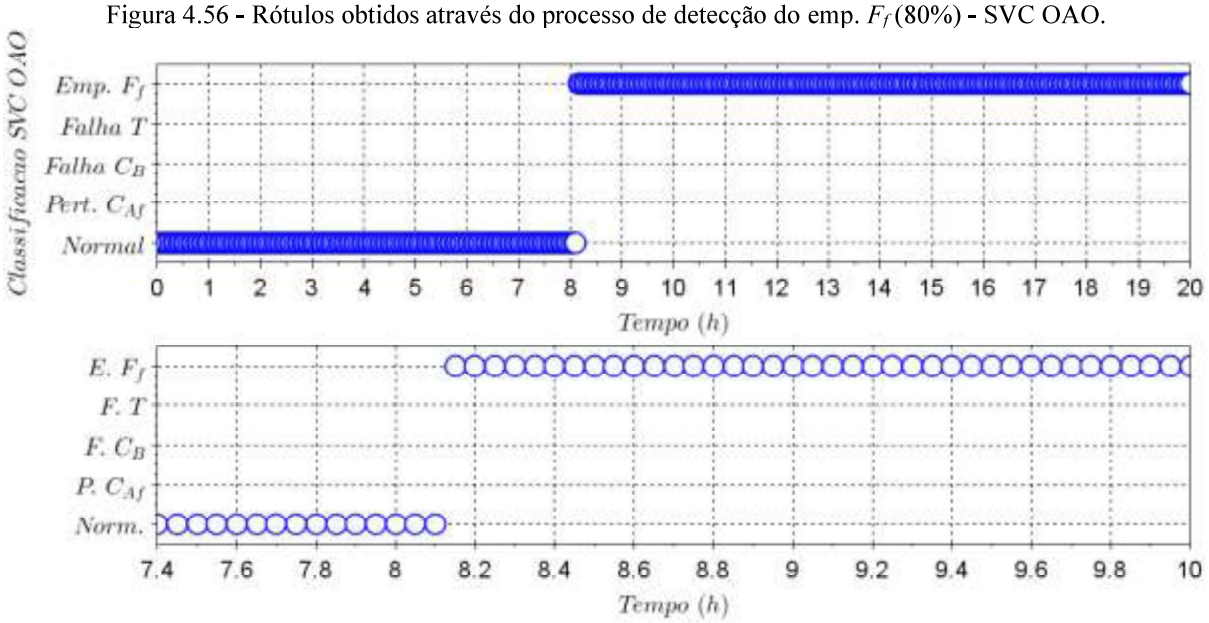


O emperramento F_f obtido através da modificação da vazão de alimentação do reator devido a um emperramento da válvula, em que a vazão passa a ser 80% da vazão de estado estacionário, foi detectada pelo modelo SVC OAA após um intervalo de amostragem (0,05 h).

A Figura 4.56 apresenta o resultado da detecção do emperramento F_f pelo modelo de máquina de vetores de suporte de classificação *one-against-one* quando utilizada 80% da vazão de alimentação do reator no estado estacionário, vazão maior que a utilizada no treinamento (70% da vazão no estado estacionário) do método de detecção de falhas.

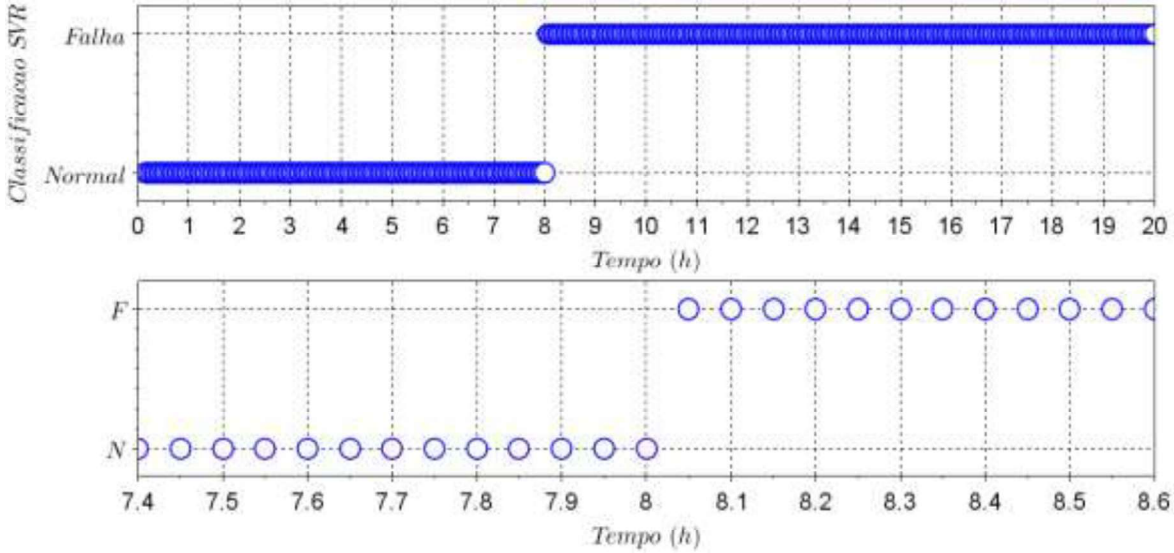
O emperramento F_f obtido através da modificação da vazão de alimentação do reator devido a um emperramento da válvula, em que a vazão passa a ser 80% da vazão de estado estacionário, foi detectada pelo modelo SVC OAO após dois intervalos de amostragem (0,10 h), mais lentamente um intervalo de amostragem (0,05 h) que o sistema detectou os dados de emperramento F_f utilizados no treinamento (70% da vazão de alimentação do reator no estado estacionário).

Figura 4.56 - Rótulos obtidos através do processo de detecção do emp. F_f (80%) - SVC OAO.



A Figura 4.57 apresenta o resultado da detecção do emperramento F_f pelo modelo de máquina de vetores de suporte de regressão quando a vazão de alimentação é maior (80% da vazão de alimentação do reator no estado estacionário) que a vazão de alimentação apresentada na Seção 4.9.3 (70% da vazão no estado estacionário).

Figura 4.57 - Rótulos obtidos através do processo de detecção do emp. F_f (80%) - SVR.



O emperramento F_f obtido através de 80% da vazão de alimentação do reator no estado estacionário foi detectado pelo modelo SVR dentro da mesma quantidade de intervalos de amostragem que os dados com 70% da vazão em estado estacionário. Para ambos os casos, a detecção ocorreu instantaneamente.

A Tabela 4.9 apresenta os instantes de detecção para as diferentes metodologias e diferentes amplitudes do emperramento F_f, assim como os índices obtidos para cada situação.

Tabela 4.9 - Índices - SVC OAA, SVC OAO e SVR - amplitudes emp. F_f.

F_f	Metodologia	TAD (h)	DF/F	DF/N	DN/N	DN/F	DF_i/F_j
60%	SVC OAA	8,10	0,9958	0,0000	1,0000	0,0042	-
	SVC OAO	8,15	0,9916	0,0000	1,0000	0,0042	0,0042
	SVR	**8,05**	**1,0000**	0,0000	1,0000	0,0000	-
70%	SVC OAA	8,10	0,9958	0,0000	1,0000	0,0042	-
	SVC OAO	8,10	0,9958	0,0000	1,0000	0,0042	-
	SVR	**8,05**	**1,0000**	0,0000	1,0000	0,0000	-
80%	SVC OAA	8,10	0,9958	0,0000	1,0000	0,0042	-
	SVC OAO	8,15	0,9916	0,0000	1,0000	0,0084	-
	SVR	**8,05**	**1,0000**	0,0000	1,0000	0,0000	-

Todos os modelos reconheceram as diferentes amplitudes do emperramento F_f. Entre as metodologias SVM de classificação, a SVC *one-against-all* detectou o emperramento F_f mais rapidamente que a metodologia SVC *one-against-one*. O modelo SVC OAO foi o único a classificar erroneamente durante um intervalo de amostragem a falha C_B ao invés do emperramento F_f quando utilizada 60% da vazão de alimentação do reator no estado estacionário. Entre as diferentes metodologias, a SVR foi a mais rápida a detectar uma operação anormal, detectando instantaneamente a falha.

4.10. Estudo de Caso - Planta Química 2

Nessa seção serão apresentados os resultados do estudo de caso da planta química apresentada em Stewart e colaboradores (2011). A metodologia de simulação e obtenção dos dados do processo foram apresentados no Capítulo 3.

4.10.1. Máquina de Vetores de Suporte de Classificação *One-Against-One*

Para a máquina de vetores de suporte de classificação (*Support Vector Classification*, SVC), na fase de treinamento e validação do modelo SVC OAO, é necessário normalizar todos os dados de treinamento e para cada grupo de dados atribuir um rótulo. Neste caso, foram atribuídos rótulos de 1 a 26 às diferentes situações (cada falha em cada variável e cada um dos dois estados estacionários). Para o SVC OAO, foram utilizadas as variáveis controladas e manipuladas de cada tanque: H_1, H_2, H_3, T_1, T_2, T_3, F_{f1}, F_{f2}, F_R, Q_1, Q_2 e Q_3.. Para os dados de treinamento, foram utilizados os referenciais apresentados no Tópico 3.6.1, cujos sinais das

variáveis controladas foram obtidos através de simulação. Do total de dados obtidos, foram utilizados dois terços dos dados simulados para o treinamento e o restante, um terço, foram utilizados para a validação do modelo, de acordo com a metodologia apresentada no Capítulo 3.

O sistema de detecção através do SVC utiliza as informações treinadas para categorizar as falhas e as operações normais. Independentemente do instante em que a falha seja aplicada, o sistema de detecção tem a capacidade de categorizar cada instante da operação do processo.

Uma vez obtido o modelo SVC OAO com precisão de 98,7% quando comparados os vetores de saída dos dados de validação com os dados do modelo obtido com os dados de treinamento, foi realizada a identificação dos sinais simulados para a detecção e diagnóstico de falhas. O SVC OAO separou 189 vetores de suporte do total de 2080 dados de treinamento como os respectivos vetores de suporte de classificação *one-against-one* das vinte e seis classes (dois estados estacionários, 1 e 2, e vinte e quatro falhas, 1 a 24). Os melhores valores de C e para o modelo SVC OAO treinado foi encontrado através de validação cruzada.

Para a detecção, foram desenvolvidos vinte e seis sinais em que ocorria transição do estado estacionário 1 para o estado estacionário 2 e vice-versa, em grupos de dois, um par para os estados estacionários e os outros 24 para cada uma das falhas, de forma a conseguir detectar de maneira satisfatória e rápida essas transições.

4.10.2. Máquina de Vetores de Suporte de Classificação *One-Against-All*

Para a máquina de vetores de suporte de classificação *one-against-all*, na fase de treinamento e validação do modelo SVC OAA, é necessário também normalizar todos os dados de treinamento e para cada grupo de dados atribuir um rótulo. Neste caso, foi atribuído rótulo 1 ao estado estacionário 1 (*ee*1) e rótulo 2 ao estado estacionário 2 (*ee*2).

Uma vez atribuídos os rótulos, deve-se normalizar todos os dados. Para o SVC OAA, foram utilizadas as variáveis controladas e manipuladas de cada tanque: H_1, H_2, H_3, T_1, T_2, T_3, F_{f1}, F_{f2}, F_R, Q_1, Q_2 e Q_3. Para os dados de treinamento, foram utilizados os referenciais apresentados no Tópico 3.6.1, cujos sinais das variáveis controladas foram obtidos através de simulação.

Do total de dados obtidos, foram utilizados dois terços dos dados simulados para o treinamento e o restante, um terço, foram utilizados para a validação do modelo, de acordo com a metodologia apresentada no Capítulo 3.

O sistema de detecção através do SVC utiliza as informações treinadas para categorizar as falhas e as operações normais. Independentemente do instante em que a falha seja aplicada, o sistema de detecção tem a capacidade de categorizar cada instante da operação do processo.

Uma vez obtido o modelo SVC OAA com precisão de 100% quando comparados os vetores de saída dos dados de validação com os dados do modelo obtido com os dados de treinamento, foi realizada a identificação dos sinais simulados para a detecção e diagnóstico de falhas. O SVC OAA separou 52 vetores de suporte do total de 726 dados de treinamento como os respectivos vetores de suporte de classificação *one-against-one* das vinte e seis classes (estados estacionários 1 e 2 e falhas 1 a 24). Os melhores valores de C e para o modelo SVC OAO treinado foi encontrado através de validação cruzada.

Para a detecção, foram desenvolvidos vinte e seis sinais em que ocorre transição do estado estacionário 1 para o estado estacionário 2 e vice-versa, em grupos de dois, um par para os estados estacionários e os outros 24, um para cada uma das falhas, de forma a conseguir detectar de maneira satisfatória e rápida essas transições.

A Tabela 4.10 apresenta a matriz de confusão normalizada do SVC OAO e do SVC OAA, que são iguais, para o estudo de caso 2.

Tabela 4.10 - Matriz de confusão normalizada - SVC OAO e SVC OAA

1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0,99	0,01	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

O Apêndice A apresenta os resultados para a detecção de falhas utilizando o método SVC OAO para todos os cenários do estudo de caso 2. A Tabela 4.11 apresenta os valores dos índices apresentados na Seção 3.3.

Tabela 4.11 - Índices - Estudo de Caso 2 - SVC OAO e SVC OAA.

Oper.	Índices	SVC OAO	SVC OAA	Oper.	Índices	SVC OAO	SVC OAA
ee1	DF/F	1,0000	1,0000	*ee2*	DF/F	1,0000	1,0000
	DF/N	0,0000	0,0000		DF/N	0,0000	0,0000
	DN/N	1,0000	1,0000		DN/N	1,0000	1,0000
	DN/F	0,0000	0,0000		DN/F	0,0000	0,0000
$H1_1$	DF/F	1,0000	1,0000	$Ff1_1$	DF/F	1,0000	1,0000
	DF/N	0,0000	0,0000		DF/N	0,0000	0,0000
	DN/N	1,0000	1,0000		DN/N	1,0000	1,0000
	DN/F	0,0000	0,0000		DN/F	0,0000	0,0000
$H1_2$	DF/F	1,0000	1,0000	$Ff1_2$	DF/F	1,0000	1,0000
	DF/N	0,0000	0,0000		DF/N	0,0000	0,0000
	DN/N	1,0000	1,0000		DN/N	1,0000	1,0000
	DN/F	0,0000	0,0000		DN/F	0,0000	0,0000
$H2_1$	DF/F	1,0000	1,0000	$Ff2_1$	DF/F	1,0000	1,0000
	DF/N	0,0000	0,0000		DF/N	0,0000	0,0000
	DN/N	1,0000	1,0000		DN/N	1,0000	1,0000
	DN/F	0,0000	0,0000		DN/F	0,0000	0,0000
$H2_2$	DF/F	1,0000	1,0000	$Ff2_2$	DF/F	1,0000	1,0000
	DF/N	0,0000	0,0000		DF/N	0,0000	0,0000
	DN/N	1,0000	1,0000		DN/N	1,0000	1,0000
	DN/F	0,0000	0,0000		DN/F	0,0000	0,0000
$H3_1$	DF/F	1,0000	1,0000	F_{R1}	DF/F	1,0000	1,0000
	DF/N	0,0000	0,0000		DF/N	0,0000	0,0000
	DN/N	1,0000	1,0000		DN/N	1,0000	1,0000
	DN/F	0,0000	0,0000		DN/F	0,0000	0,0000
$H3_2$	DF/F	1,0000	1,0000	F_{R2}	DF/F	1,0000	1,0000
	DF/N	0,0000	0,0000		DF/N	0,0000	0,0000
	DN/N	1,0000	1,0000		DN/N	1,0000	1,0000
	DN/F	0,0000	0,0000		DN/F	0,0000	0,0000
$T1_1$	DF/F	1,0000	1,0000	$Q1_1$	DF/F	1,0000	1,0000
	DF/N	0,0000	0,0000		DF/N	0,0000	0,0000
	DN/N	1,0000	1,0000		DN/N	1,0000	1,0000
	DN/F	0,0000	0,0000		DN/F	0,0000	0,0000
$T1_2$	DF/F	1,0000	1,0000	$Q1_2$	DF/F	1,0000	1,0000
	DF/N	0,0000	0,0000		DF/N	0,0000	0,0000
	DN/N	1,0000	1,0000		DN/N	1,0000	1,0000
	DN/F	0,0000	0,0000		DN/F	0,0000	0,0000
$T2_1$	DF/F	1,0000	1,0000	$Q2_1$	DF/F	1,0000	1,0000
	DF/N	0,0000	0,0000		DF/N	0,0000	0,0000
	DN/N	1,0000	1,0000		DN/N	1,0000	1,0000
	DN/F	0,0000	0,0000		DN/F	0,0000	0,0000
$T2_2$	DF/F	1,0000	1,0000	$Q2_2$	DF/F	1,0000	1,0000
	DF/N	0,0000	0,0000		DF/N	0,0000	0,0000
	DN/N	1,0000	1,0000		DN/N	1,0000	1,0000
	DN/F	0,0000	0,0000		DN/F	0,0000	0,0000
$T3_1$	DF/F	1,0000	1,0000	$Q3_1$	DF/F	1,0000	1,0000
	DF/N	0,0000	0,0000		DF/N	0,0000	0,0000
	DN/N	1,0000	1,0000		DN/N	1,0000	1,0000
	DN/F	0,0000	0,0000		DN/F	0,0000	0,0000
$T3_2$	DF/F	1,0000	1,0000	$Q3_2$	DF/F	1,0000	1,0000
	DF/N	0,0000	0,0000		DF/N	0,0000	0,0000
	DN/N	1,0000	1,0000		DN/N	1,0000	1,0000
	DN/F	0,0000	0,0000		DN/F	0,0000	0,0000

No Apêndice A, da Figura A.1 até a Figura A.52 estão relacionados os resultados de ambas as metodologias SVC OAO e SVC OAA.

4.11. Comentários

Analisando a aplicação de diferentes amplitudes de falhas, pode-se afirmar que as máquinas de vetores de suporte, de classificação e de regressão, são capazes de detectar falhas e padrões mesmo que não tenham sido treinadas com a mesma amplitude da falha amostrada. Nos casos em que a amplitude do padrão a ser detectado é maior que a amplitude dos dados de treinamento, as metodologias são capazes de detectar mais rapidamente as diferentes falhas.

O método de separação de dados através de hiperplanos apresenta grandes possibilidades na aplicação de detecção e diagnóstico de falhas. Quanto menores as amplitudes das falhas treinadas, melhor a capacidade do método SVM classificar diferentes falhas.

No próximo capítulo, é apresentada teoria de redes neuronais artificiais, suas aplicações em detecção e diagnóstico de falhas, e os resultados da detecção de dados para os estudos de casos.

CAPÍTULO 5
DETECÇÃO DE FALHAS COM REDES NEURONAIS ARTIFICIAIS

5.1. Introdução

Redes neuronais artificiais são sistemas computacionais estruturados de maneira similar aos neurônios e às estruturas do cérebro, em que existem ligações por onde passam as informações. Os neurônios, ou nós simples, são interligados formando uma rede de neurônios. A propriedade mais importante das redes neuronais é a habilidade de aprender do ambiente, melhorando assim o seu desempenho. Essa aprendizagem é realizada através de um processo iterativo de ajustes aplicado aos pesos de cada neurônio, sendo este procedimento denominado como treinamento. O aprendizado ocorre assim que a rede neuronal atinja uma solução generalizada para uma classe de problemas.

5.2. Aplicações de Redes Neuronais Artificiais

As redes neuronais artificiais (RNA) são consideradas como uma das técnicas de inteligência artificial e foram desenvolvidas para emular os neurônios biológicos humanos, com a sua aplicação pode se generalizar comportamentos a partir de informações fornecidas através de amostras. As RNAs têm demonstrado um bom desempenho na modelagem de processos não lineares e para controle de processos, sendo a sua aplicação extensa em muitas áreas das engenharias: craqueamento catalítico, processos de polimerização, petroquímica, desenvolvimento de novos materiais, detecção de falhas, processos siderúrgicos, processos de tratamento de efluentes industriais, etc (GALUSHKIN, 2007).

A ampla difusão de aplicações foi devido às seguintes características atrativas das redes neuronais (ASSIDJO *et al*., 2006):

- As RNAs têm habilidade para aproximar funções não lineares arbitrárias.
- Podem ser treinadas facilmente usando dados registrados do sistema em estudo.
- São facilmente aplicáveis para sistemas multivariáveis.
- Não requerem especificação da relação estrutural entre dados de entrada e saída.

As RNAs podem ser classificadas em dois grandes grupos: supervisionadas e não-supervisionadas.

As RNAs supervisionadas são redes em que o treinamento ocorre através do fornecimento de dados da entrada e da saída da rede, ou seja, a RNA é dita supervisionada se a saída desejada é conhecida. Durante o processo de treinamento, os valores de saída obtidos

através da rede são comparados com os valores desejados. Dependendo da diferença entre a saída real e a desejada, calcula-se o erro. A rede então é atualizada época após época, de acordo com uma taxa de aprendizagem, até o menor erro possível ser obtido ou o número de épocas definido para o treinamento ser extrapolado.

As RNAs não-supervisionadas, a partir dos dados de entrada da rede, são treinadas com dados de entrada, porém sem dados de saída ou rótulos que supervisionariam o treinamento da rede. Ao invés de estabelecer a saída da rede neuronal são indicadas em quantas classes diferentes os dados serão separados. Não é possível nesse tipo de RNA determinar como será o resultado do processo de aprendizagem. Durante o processo de treinamento, as unidades (pesos) nessas RNAs são "arranjados" dentro de uma certa gama, dependendo dos valores das entradas fornecidas. O objetivo é agrupar unidades similares em certas áreas das faixas de valores. Esse tipo de efeito pode ser utilizado efetivamente para propósitos de classificação de padrões.

A seguir é apresentada uma revisão dos fundamentos teóricos das RNAs para a sua utilização na modelagem empírica e controle de processos.

5.3. Conceito e Características

Segundo Haykin (2007) uma RNA é um processador massivamente paralelo constituído por unidades de processamento simples, que têm a propensão natural para armazenar conhecimento experimental e torná-lo disponível para o uso. Ela se assemelha ao cérebro em dois aspectos.

1) O conhecimento é adquirido pela rede a partir de seu ambiente através de um processo de aprendizagem.
2) Forças de conexão entre unidades de processamento de informação (neurônios) conhecidas como pesos sinápticos são utilizadas para armazenar o conhecimento adquirido.

O conceito salienta como características importantes da rede, sua estrutura, com unidades de processamento simples e interconectadas, e suas capacidades de: processamento paralelo da informação, aquisição de conhecimento por aprendizagem, armazenamento e disponibilização para uso futuro. Segundo Hinton (2001), a propriedade mais interessante das redes neuronais é sua habilidade de aprender com exemplos através da adaptação dos pesos das conexões.

Além das capacidades mencionadas no conceito, a rede tem capacidade de generalizar a informação aprendida, entendendo por generalização o fato de que a rede neuronal pode produzir saídas adequadas para entradas que não estavam presentes durante o treinamento (aprendizagem).

Na literatura são reconhecidos os benefícios da utilização das redes neuronais como resultado das capacidades acima mencionadas. Gomes e Ludermir (2008) manifestam que as redes neuronais são ferramentas úteis para o reconhecimento de padrões ou classificação de dados mesmo na presença de ruídos ou dados incompletos.

Haykin (2007) menciona como propriedades relevantes que oferece a utilização das redes neuronais:

- *Não linearidade:* propriedade necessária para modelagem de mecanismos físicos gerando sinal de entrada inerentemente não linear.
- *Mapeamento de entrada-saída:* a aprendizagem supervisionada envolve a modificação dos pesos sinápticos de uma rede neuronal pela aplicação de um conjunto de dados amostrais do problema. Cada amostra consiste de um sinal de entrada único e de uma resposta desejada correspondente. Assim, a rede neuronal aprende através dos exemplos a construir um mapeamento de entrada-saída para a resolução do problema desejado.
- *Adaptabilidade:* a rede neuronal pode ser facilmente treinada novamente para lidar com pequenas modificações nas condições operacionais do meio ambiente. Além disso, quando é necessária a sua operação em um ambiente não estacionário, uma rede neuronal pode ser projetada para modificar seus pesos sinápticos em tempo real.
- *Resposta a evidências:* no contexto da classificação de padrões, a rede neuronal pode ser utilizada para rejeitar padrões ambíguos, caso eles estejam presentes, e com isso melhorar o desempenho de classificação da rede.
- *Informação contextual:* cada neurônio da rede neuronal é afetado pela atividade de todos os outros neurônios na rede, consequentemente, a informação contextual é tratada naturalmente pela rede neuronal.
- *Tolerância a falhas:* uma rede neuronal, implementada na forma física (em *hardware*), tem o potencial de ser inerentemente tolerante a falhas, ou capaz de realizar computação robusta, no sentido de que seu desempenho se degrada suavemente sob condições de operações adversas.
- *Implementação em VLSI (Very Large Scale Integration):* a natureza paralela de uma rede neuronal a torna potencialmente rápida na computação de certas tarefas. Esta mesma característica torna uma rede neuronal adequada para implementação utilizando tecnologia de integração em escala muito ampla.

- *Uniformidade de análise e projeto:* basicamente, as redes neuronais desfrutam de universalidade como processadores de informação. A mesma notação é utilizada em todos os domínios envolvendo a aplicação de redes neuronais.
- *Analogia neurobiológica:* o cérebro é uma prova viva de que o processamento paralelo tolerante a falhas não é somente fisicamente possível, mas também rápido e poderoso.

Quanto ao funcionamento das redes neuronais, cada neurônio em uma rede processa os sinais que recebe e envia os sinais processados para outros neurônios. Em uma estrutura usual de rede neuronal artificial os neurônios estão interconectados e distribuídos em camadas. O objetivo de uma rede neuronal é processar a informação de acordo com seu prévio treinamento com dados de entrada-saída. No processamento de informação as redes neuronais são paralelas, suas numerosas operações são executadas simultaneamente (SIMÕES; SHAW, 2007).

Os modelos de redes neuronais podem ser definidos como referência às seguintes características: o conjunto de unidades computacionais, as funções de ativação das unidades computacionais, a topologia da rede, o processo de aprendizado da rede e a interação com o ambiente externo que fornece informação para a rede e/ou interage com ela.

As unidades computacionais ou neurônios têm como componentes básicos: um conjunto de sinapses, um somatório e uma função de ativação (HAYKIN, 2007).

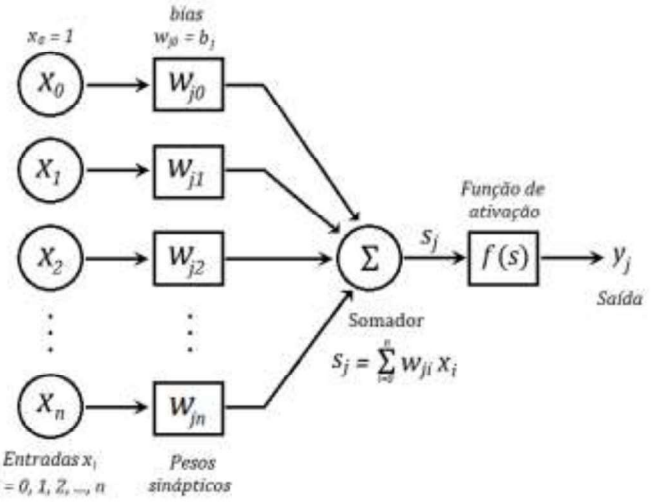
As sinapses ou elos de conexão são as conexões entre neurônios, cada uma caracterizada por um peso sináptico. O peso sináptico w_{ji} caracteriza a conexão entre a saída do neurônio i e a entrada do neurônio j.

O somador ou combinador linear calcula a soma ponderada de cada sinal de entrada x_i multiplicada pelo seu peso sináptico w_{ji}. O modelo de neurônio inclui também um parâmetro de ajuste para o resultado da combinação linear que é chamado de *bias*, o b_j é considerado equivalente a um peso sináptico com sinal de entrada igual a 1.

A função de ativação que recebe o resultado do somador e restringe a amplitude da saída de um neurônio a um intervalo permissível. Tipicamente, o intervalo normalizado da amplitude da saída de um neurônio é $[0,1]$ ou alternativamente $[-1,1]$.

A Figura 5.1 descreve um neurônio artificial, $x_1, x_2, \cdots, x_n$, são os sinais de entrada, $w_{j1}, w_{j2}, \cdots, w_{jn}$ são os pesos sinápticos do neurônio j, s_j é a saída do combinador linear devido aos sinais de entrada; b_j é o *bias*; $f(s)$ é a função de ativação; y_j é o sinal de saída do neurônio j.

Figura 5.1 - Modelo de um neurônio.



Fonte: adaptado de MCCULLOCH; PITTS (1943).

5.4. Tipos de Funções de Ativação

A função de ativação transforma o sinal de entrada em um sinal de saída. A escolha da função de ativação é considerada por muitos especialistas tão importante quanto a arquitetura ou o algoritmo de aprendizagem da rede neuronal (GOMES; LUDERMIR, 2008). As funções de ativação mais utilizadas na construção de redes neuronais artificiais são: função sigmoide, função linear, função de limiar e a função gaussiana.

Função sigmoide

A função sigmoide é definida como uma função estritamente crescente, contínua e diferenciável em todos os pontos, tem aproximação assintótica a seus valores de saturação e mostra um equilíbrio adequado entre comportamento linear e não linear (HAYKIN, 2007; MENNON *et al*., 2000). São alguns exemplos a função sigmoide "logística" e a tangente hiperbólica. Segundo Mennon e colaboradores (1996), em poucos casos as curvas sigmoides podem ser descritas por equações contínuas. O gráfico da função sigmoide tem formato em S.

Função sigmoide logística

É a função de ativação mais popular utilizada em redes neuronais (MENNON *et al*., 2000). Pode ser definida de acordo com a Equação (5.1).

$$f(s) = \frac{1}{1 + \exp(-as)} \tag{5.1}$$

em que s é a entrada da rede para o neurônio, $f(s)$ é a saída do neurônio e a é o parâmetro de inclinação da função sigmoide. A saída do neurônio com função de ativação logística varia

entre 0 e 1.

Função tangente hiperbólica

Pode ser definida de acordo com a Equação (5.2).

$$f(s) = \tanh(s) = \frac{1 - \exp(-as)}{1 + \exp(-as)} \tag{5.2}$$

em que $f(s)$ é a saída do neurônio que varia entre -1 e +1.

As funções de ativação sigmoides são usadas tradicionalmente em redes *feedforward* com aprendizagem *backpropagation*. Segundo Haykin (2007) se a função de ativação sigmoide incorporada no modelo do neurônio da rede é antissimétrica (por exemplo, a tangente hiperbólica), a rede pode, em geral, aprender mais rápido (em termos do número de iterações, ou épocas, necessárias para o treinamento) do que quando ela é não-simétrica (por exemplo, a função logística padrão).

Função linear

Representa uma equação linear como apresentada na Equação (5.3).

$$f(s) = as \tag{5.3}$$

em que a é um número real que define a saída linear para os valores de entrada.

A função linear também pode ser definida por segmentos dentro de um intervalo, nesse caso se conhecem como funções lineares definidas por partes (*piecewise linear functions*) e consistem de um número finito de segmentos lineares. As funções de limiar e de rampa são casos especiais dessas funções. As funções lineares definidas por partes são mais fáceis de calcular do que as funções não lineares, tais como as funções sigmoides, e têm sido usadas como aproximações das mesmas (MENNON *et al*., 2000).

Um caso de função linear definida por partes é apresentado na Equação (5.4).

$$f(s) = \begin{cases} 1 & \text{se} \quad s \geq +1/2 \\ as & \text{se} \quad -1/2 < s < +1/2 \\ -1 & \text{se} \quad s \leq -1/2 \end{cases} \tag{5.4}$$

em que $f(s)$ tem três segmentos de linha reta, o segmento intermediário tem coeficiente angular a.

Função de limiar

Para uma saída do neurônio $f(s)$ entre 0 e 1, a função de limiar é expressa como a Equação (5.5).

$$f(s) = \begin{cases} 1 & \text{se} \quad s \geq 0 \\ 0 & \text{se} \quad s < 0 \end{cases} \tag{5.5}$$

Neste caso, a saída de um neurônio assume o valor 1, se a entrada é não negativa, e 0, em caso contrário.

Quando se deseja que a saída do neurônio $f(s)$ esteja entre -1 e +1, a função de limiar pode ser expressa como a Equação (5.6).

$$f(s) = \begin{cases} +1 & se \quad s > 0 \\ 0 & se \quad s = 0 \\ -1 & se \quad s < 0 \end{cases} \tag{5.6}$$

O modelo de neurônio de McCulloch e Pitts (1943) é uma unidade lógica tipo limiar (*Threshold Logic Unit*, TLU) com entradas binárias (0 ou 1) e uma saída binária, os pesos associados a cada entrada são -1 ou +1.

Função gaussiana

A saída do neurônio é expressa como a Equação (5.7).

$$f(s) = \exp\left(\frac{-(x - C_i)^2}{\tau_i}\right) \tag{5.7}$$

em que C_i é um vetor que determina um centro associado e τ_i é um escalar que representa o "espalhamento" da função de ativação da unidade. Dessa maneira, a saída da unidade é máxima quando $x = C_i$ e decai suavemente na medida em que x se afasta do centro de C_i.

A função de ativação gaussiana é utilizada em redes neuronais RBF (redes de funções de base radial).

Na literatura existem estudos que mostram aplicações específicas de outras funções de ativação para o aprendizado da rede neuronal, como descrito a seguir.

Hornik (1991) mostrou que funções de ativação contínuas, limitadas e inconstantes permitem que as redes *feedforward* multicamadas com apenas uma camada escondida podem ser aproximadores universais de funções.

Leshno e colaboradores (1993) mostraram utilidades das funções de ativação contínuas definidas por partes (*piecewise linear functions*) no desempenho das redes *feedforward* multicamadas na aproximação de funções.

Guarnieri e colaboradores (1999) e Vecci e colaboradores (1998) relataram que em redes neuronais *feedforward* multicamadas, com o uso de uma função de ativação *spline* adaptativa baseada na *spline* cúbica de Catmull-Rom como função de ativação dos neurônios, é possível ter melhorias em termos de capacidade de generalização e de velocidade do aprendizado em ambas as tarefas de reconhecimento de padrões e processamento de dados.

Gomes e Ludermir (2008) utilizaram uma rede *Perceptron* multicamada (redes neuronais artificiais *feedforward* multicamadas) para aproximar funções, com as inversas das funções de ligação *complemento log-log* e *probit* como funções de ativação e compararam seu uso com o uso de funções de ativação sigmoide logística. *Probit* são funções de regressão em que a variável dependente pode somente assumir dois valores, como *sim* ou *não*. Segundo os autores, o uso das funções propostas é mais adequado em casos específicos de dados que seguem uma distribuição binomial e em casos de presença de observações extremas.

Sharma e Chandra (2010) estudaram a aplicação de redes neuronais construtivas a problemas de regressão e verificaram o papel da função de ativação sigmoide logística com inclinação adaptativa em redes neuronais *feedforward* construtivas para um melhor desempenho em generalização e menor tempo de treinamento.

5.5. Redes Neuronais Artificiais Supervisionadas

Existem três tipos principais de estratégias de treinamento de RNAs: o treinamento (ou aprendizado) supervisionado, o treinamento por reforço e o treinamento não-supervisionado. No aprendizado supervisionado, a RNA é treinada com auxílio de um "supervisor". A rede deve possuir pares de entrada e saída (conjunto de entradas e conjunto de saídas desejadas para cada entrada), organizados em matrizes de dados entrada/saída, também conhecida como tabela atributos/valores, de maneira que toda vez que for apresentada à rede uma entrada, deverá ser verificado se a saída obtida através dos cálculos efetuados através dos pesos sinápticos da rede, confere com a saída desejada para aquela entrada; se for diferente, a rede deverá ajustar os pesos de modo a armazenar o conhecimento desejado. O treinamento se processará até que todo o conjunto de treinamento (entradas e saídas) seja satisfeito, com uma taxa de acerto dentro de uma faixa considerada satisfatória. O "supervisor" na realidade é a própria matriz de dados que ensina à rede qual a resposta correta para cada entrada apresentada, sendo este um conhecimento indutivo (KRÖSE; VAN DER SMAGT, 1996; MEHROTRA *et al*., 1996; HAYKIN, 2005; HAYKIN, 2007; DA SILVA *et al*., 2010).

O aprendizado por reforço é considerado uma variação do treinamento supervisionado. Não existe a presença do "supervisor", mas de um "observador crítico externo" ou rede crítica, que através de uma função custo avalia a resposta fornecida pela rede, com a finalidade de prever o reforço e aplicar um processo de aprendizagem que adapta os pesos sinápticos de maneira a obter um mapeamento que minimiza o valor da função custo. O algoritmo de treinamento, baseado em métodos estocásticos, ajusta os pesos dos neurônios baseado em informações qualitativas ou quantitativas inferidas do sistema mapeado que são usadas para medição e correção do desempenho de aprendizado. O treinamento é, portanto, por tentativa e erro; se a resposta for satisfatória, recompensa-se o neurônio pelo incremento de seu peso sináptico e limiar; se for insatisfatória, pune-se o neurônio (KRÖSE; VAN DER SMAGT, 1996; SUTTON; BARTO, 1998; DA SILVA *et al*., 2010).

No aprendizado não-supervisionado não existe um conjunto de saídas desejadas, portanto não há "supervisores", uma vez que não existem dados suficientes que indiquem uma resposta desejada para um dado padrão de entrada. Esse treinamento também é conhecido como auto supervisionado ou auto organizado, já que a própria rede se auto organiza em relação às particularidades encontradas entre os elementos do subconjunto amostra, identificando subconjuntos que contêm similaridades (*clusters*). O aprendizado ocorre pela descoberta de padrões ou relevâncias dentro do grupo de dados pela rede ao longo do treinamento, de forma que os dados são agrupados (clusterizados) conforme esses padrões. Dependendo do conhecimento do problema por parte do projetista de rede e também do tipo de algoritmo não-supervisionado escolhido, o número de *clusters* pode ser previamente determinado (KRÖSE; VAN DER SMAGT, 1996; HAYKIN, 2005; HAYKIN, 2007; DA SILVA *et al*., 2010)

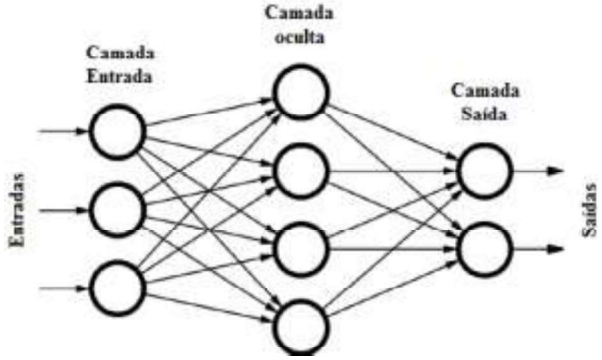
As RNAs supervisionadas podem ser classificadas como RNAs de alimentação direta, RNAs retroalimentadas e *perceptrons* de múltipla camada, entre outros tipos de algoritmos supervisionados.

5.5.1. Alimentação direta ou *feedforward*

A rede neuronal é separada em camadas, em que todas as unidades enviam suas saídas apenas para as unidades situadas na próxima camada. Estas redes não possuem realimentação local e não existem laços (NASCIMENTO; YONEYAMA, 2008).

A rede neuronal é dita totalmente conectada quando cada unidade de uma camada da rede está conectada a todas as unidades da camada adjacente seguinte. A rede é parcialmente conectada se algumas das conexões estiverem faltando na rede neuronal (HAYKIN, 2007) (Figura 5.2).

Figura 5.2 - Rede Neuronal *Feedforward* - uma camada oculta.



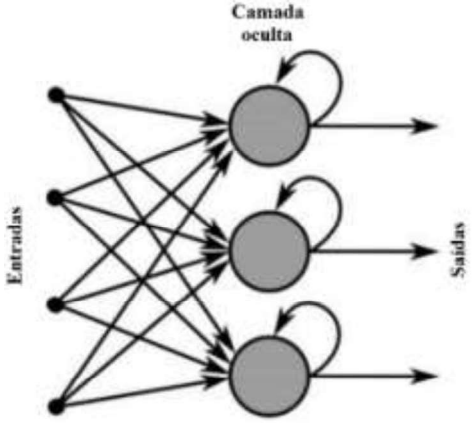
Fonte: adaptado de MSA TECHNOSOFT, 2018.

Em uma rede neuronal em camadas, a mais simples é a que tem uma camada de nós de entrada e uma camada de neurônios de saída (nós computacionais). As redes com múltiplas camadas possuem uma ou mais camadas ocultas. Mais simplificadamente, uma rede *feedforward* com i entradas, h_1 neurônios na primeira camada oculta, h_2 neurônios na segunda camada oculta e o neurônios na camada de saída é referida como uma rede $i: h_1: h_2: o$.

5.5.2. Retroalimentada ou *feedback*

Uma rede neuronal retroalimentada ou *feedback* se distingue de uma rede neuronal *feedforward* por ter pelo menos um laço de retroalimentação local (KOLEN; KREMER, 2001). Segundo Nascimento e Yoneyama (2008), uma rede tipo *feedback* é em geral um sistema dinâmico não linear em que a estabilidade da rede é um tópico de grande importância. Mandic e Chambers (2001) apontam que as conexões recorrentes na rede neuronal podem torná-la mais instável e mais sensível ao ruído. Braga e colaboradores (2000) manifestam que as redes neuronais retroalimentadas são mais apropriadas para resolver problemas que envolvem processamento dependente do tempo (Figura 5.3).

Figura 5.3 - Rede Neuronal *Feedback* ou Retroalimentada - uma camada oculta.

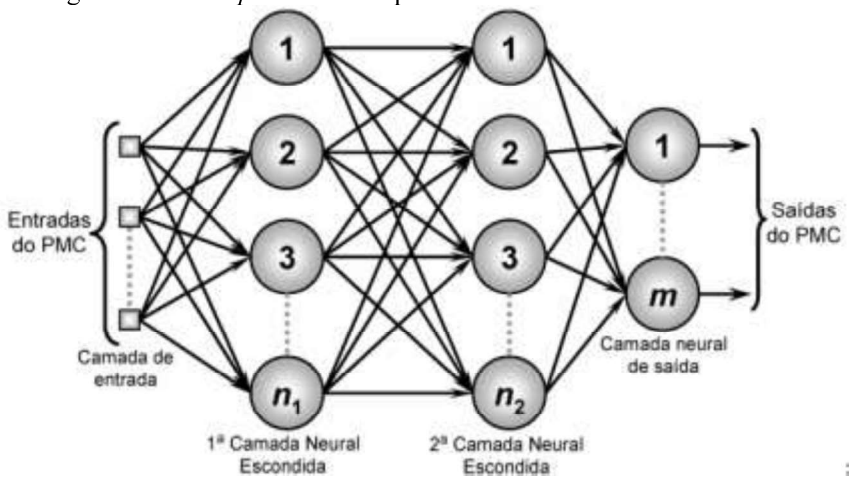


Fonte: adaptado de DONGES, 2018.

5.5.3. *Perceptrons* de múltiplas camadas

São redes neuronais artificiais de múltiplas camadas de alimentação direta (*feedforward*), consistem de um conjunto de unidades sensoriais que constituem a camada de entrada, e neurônios (unidades computacionais) distribuídos em uma ou mais camadas ocultas e uma camada de saída. Os sinais de entrada se propagam para frente através da rede camada por camada. Segundo Haykin (2007), essas redes têm sido aplicadas com sucesso para resolver diversos problemas difíceis através de seu treinamento de forma supervisionada com o algoritmo de retropropagação do erro (*backpropagation*). Hornik e colaboradores (1989) afirmam que as redes neuronais *feedforward* de múltiplas camadas são aproximadores universais, porque possuem a capacidade de aproximar qualquer tipo de função mensurável com um grau desejado de precisão, em que qualquer insucesso na aplicação é devido à aprendizagem inadequada, número insuficiente de unidades ocultas ou a falta de uma relação determinística entre a entrada e a saída desejada. A capacidade de aproximação de funções dessas redes é também demonstrada através da prova do Teorema de Cybenko (1989) (Figura 5.4).

Figura 5.4 - *Perceptron* de Múltiplas Camadas - duas camadas ocultas.



Fonte: BATISTA, 2012.

Denomina-se algoritmo de aprendizado o conjunto de regras definidas para a solução de um problema de aprendizado. Outro fator importante é de que forma uma rede neuronal se relaciona com o ambiente. No contexto existem os seguintes modelos de aprendizado:

- Aprendizado Supervisionado: existe um agente externo que indica à rede a resposta desejada para o padrão de entrada.
- Aprendizado Não Supervisionado (auto-organização): não existe um agente externo que indica à rede a resposta desejada para os padrões de entrada.
- Reforço: um crítico externo avalia a resposta fornecida pela rede.

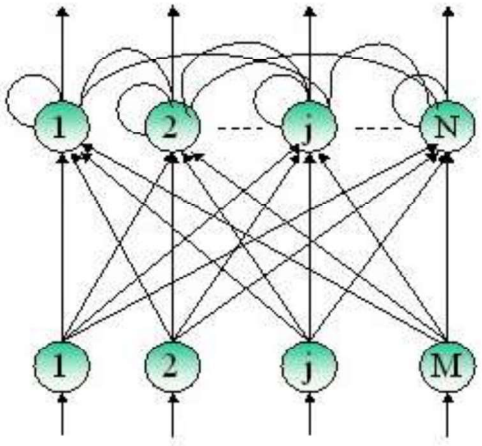
As aplicações mais comuns das redes neuronais são em reconhecimentos automáticos de alvos, reconhecimento de caracteres, robótica, diagnóstico médico, sensoriamento remoto, processamento de voz, biometria, identificação de modelos, detecção de falhas.

5.6. Redes Neuronais Artificiais Não-Supervisionadas

O aprendizado não-supervisionado, ou aprendizado competitivo, é um método no qual os neurônios competem entre si pelo direito de responder ao estímulo de um subconjunto de dados de entrada. O aprendizado ocorre através do aumento da especialização de cada neurônio da rede em relação a subconjuntos de entradas, o que o torna bastante apropriado para encontrar *clusters* automaticamente dentro de um grupo de dados através da descoberta de características estatisticamente proeminentes nesses dados (CARPENTER; GROSSBERG, 1987).

Tomando como exemplo uma RNA com uma única camada de neurônios, o princípio básico da aprendizagem competitiva é a competição entre neurônios com o objetivo de que um deles saia vencedor (processos não-supervisionados), e cuja recompensa será o ajuste de seus pesos sinápticos proporcionalmente aos valores da entrada para que aperfeiçoe seu estado para a próxima competição, ou valores da entrada. A Figura 5.5 apresenta a topologia de uma RNA com aprendizado competitivo. As conexões laterais assumem efeito inibitório de um neurônio sobre o outro, implicando o quanto um neurônio pode influenciar sobre a resposta de saída de outro neurônio (CARPENTER; GROSSBERG, 1987; DA SILVA *et al*., 2010).

Figura 5.5 - Representação esquemática de uma RNA - aprendizado competitivo.



Fonte: YEGNANARAYANA, 2002.

A regra usual para definição do neurônio vencedor é a que determina a menor distância euclidiana entre o vetor de pesos de cada neurônio e o vetor de entrada, de um conjunto contendo elementos de n-ésimas amostras ($\mathbf{x}^n$) apresentadas na entrada da rede: aquele neurônio *j* que obtiver a menor distância frente a amostra **x** será considerado o vencedor, de acordo com

a Equação (5.8),

$$dist_j^{(n)} = \sqrt{\sum_{i=1}^{n} \left(x_i^{(n)} - w_i^j\right)^2} \tag{5.8}$$

em que $dist_j^{(n)}$ corresponde à distância euclidiana entre o vetor de entrada representando a n-ésima ($\mathbf{x}^{(n)}$) amostra em relação ao vetor de pesos do *j*-ésimo neurônio (w^j); *j* corresponde aos neurônios 1, 2, ..., *j* (COSTA *et al*., 2011).

O vetor de pesos sinápticos do neurônio é ajustado de modo a aproximá-lo cada vez mais da amostra apresentada, de acordo com a Equação (5.8), em que o vetor de pesos do neurônio vencedor é ajustado junto ao valor da taxa de aprendizagem, η, e do número de épocas, *n*.

A quantidade inicial de neurônios a ser utilizada para representar as classes com características em comum é desconhecida devido ao aprendizado não-supervisionado. Os padrões a serem classificados estão projetados no espaço de definição dos vetores de pesos, cuja dimensão é a mesma a qual as amostras de entradas são definidas (CARPENTER; GROSSBERG, 1987; DA SILVA *et al.*, 2010).

Cada vetor de peso sináptico estará posicionado no centro de um aglomerado de dados com características em comum, ou seja, um *cluster*, após o algoritmo competitivo convergir. Um *cluster* é um conjunto de dados (objetos), no qual cada dado está mais próximo (ou possui características mais similares) a dados dentro do *cluster* do que qualquer dado fora dele; *cluster* é também um conjunto de dados no qual cada dado está mais próximo ao dado, conhecido como protótipo, que reúne características que melhor representam o *cluster*. Portanto, esse protótipo possui relevância estatística para representar o conjunto de dados, correspondendo ao centroide, ou centro gravitacional do *cluster* (JAIN; DUBES, 1988).

Cada *cluster* representará sua respectiva classe e sua quantidade representada dependerá do número de neurônios na RNA. O número de *clusters* ou classes na análise de um conjunto de dados pode variar conforme a resolução exigida, seja mais "fina" ou "grosseira" (DA SILVA *et al*., 2010; JAIN; DUBES, 1988; XU; TIAN, 2015).

O ponto crucial em uma análise de *clusters* é estabelecer qual o nível de similaridade, como deve-se medi-lo, e se essa noção de similaridade depende do tipo de problema analisado. Um especialista é capaz de esclarecer com máxima precisão em que *clusters* devem ser alocados determinados dados, com ou sem aplicação de estratégias estatísticas (JAIN; DUBES, 1988).

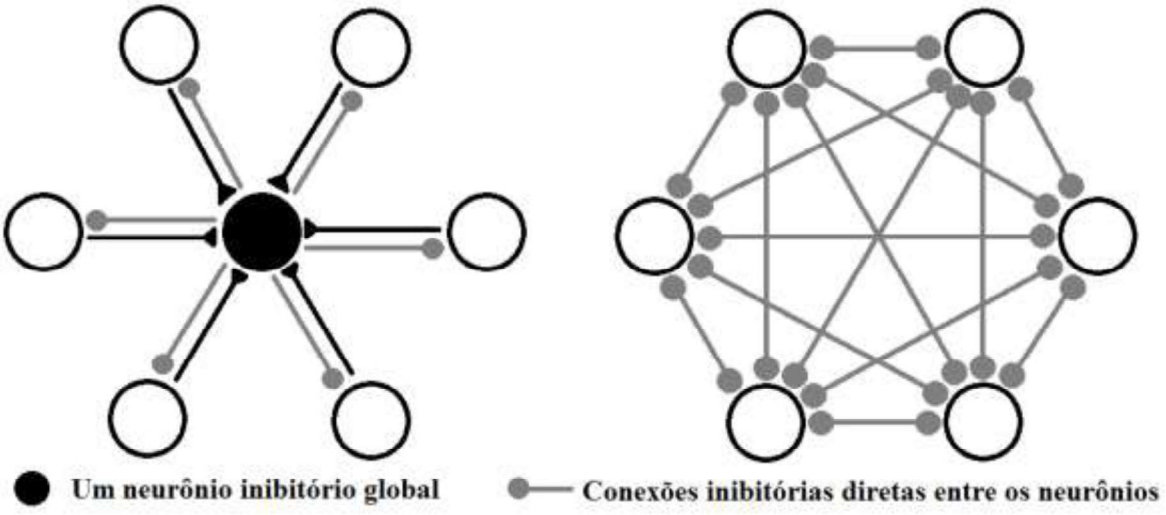
Uma vez resolvido o número de *clusters* representativos das amostras, após a convergência da rede de classificação, quando uma nova amostra for apresentada à entrada da RNA, a sua classificação é realizada verificando a proximidade ou probabilidade da nova amostra fazer parte de cada *cluster*.

Diversos algoritmos baseados em aprendizagem não-supervisionada vêm sendo desenvolvidos ao longo das últimas décadas para a análise e identificação de *clusters* em um grupo de dados. Entre esses algoritmos, pode-se listar como os mais representativos o algoritmo *winner take all* (WTA), *Frequency-sensitive Competitive Learning* (FSCL), *Rival-penalize Competitive Learning* (RPCL), *Self-Organizing Map* (SOM, mapa auto-organizável ou mapa de Kohonen) e *Neural Gas* (NG) (BALAKIN *et al.*, 1996; BUDURA *et al*., 2006; COSTA *et al*., 2011; DA SILVA *et al.*, 2010; MAASS, 2000; MALONE *et al*., 2006).

5.6.1. Winner-Take-All (WTA)

A RNA com algoritmo WTA é baseada no princípio computacional no qual os neurônios competem entre si pela ativação na camada de saída através da inibição mútua enquanto, de forma simultânea, ativam a si próprios através de conexões reflexivas (Figura 5.6). Algum tempo depois, somente o neurônio com maior estado de ativação permanecerá ativo, o que obtiver entrada mais forte, enquanto todos os outros serão desativados. A rede utiliza inibição não linear para selecionar o maior conjunto de entradas (CARPENTER; GROSSBERG, 1987; MAASS, 2000; OSTER *et al*., 2009).

Figura 5.6 - Esquema da conectividade inibitória em RNAs WTA.



No aprendizado competitivo, então, quando cada padrão de entrada é apresentado à RNA, o neurônio vencedor (*Best Matching Neuron*, BMU) é ativado na camada de saída e seu peso sináptico modificado, enquanto os demais neurônios perdedores permanecem inativos

(HAYKIN, 2007; MA; CAO, 2006; RUMELHART; ZIPSER, 1986).

As RNAs WTA possuem inspiração em processos de decisão "tudo ou nada", encontrados em neurônios corticais, ou seja, essa especificidade também é obedecida em uma rede WTA típica, em que a inibição entre neurônios excitatórios é mediada por populações de neurônios inibitórios. Os tipos de conexão podem ser modificados para ajustar a quantidade de inibição entre neurônios, modificando também a força da competição (MAASS, 2000; OSTER *et al*., 2009).

5.6.2. Frequency-Sensitive Competitive Learning (FSCL)

Uma grande desvantagem das RNAs competitivas clássicas, como K-*means* e WTA, além da incapacidade de selecionar automaticamente o número ótimo de clusters, é a subutilização de neurônios, chamados de "unidades mortas" ou conhecido também como problema do "vetor preso". Uma unidade morta é definida como um neurônio (ou nó da rede) que nunca encontra um vetor de entrada próximo a ele. O principal motivo da ocorrência de unidades mortas é que os centros dos *clusters* são inicializados de forma imprópria, existindo um centro $\boldsymbol{w}_c$ que não representa nenhum vetor de entrada, assumindo o estado de unidade morta.

O algoritmo FSCL foi criado para contornar essa desvantagem. Esse algoritmo deriva do K-*means*, com a adição de um fator de consciência, que retém as vantagens computacionais do algoritmo de origem, removendo o problema das unidades mortas. O FSCL considera com qual frequência cada unidade neural é vencedora em uma competição, e essa informação é considerada para assegurar que durante o processo de treinamento todas as unidades neurais sejam modificadas aproximadamente o mesmo número de vezes (AHALT *et al*., 1990; AHALT; FOWLER, 1993; GALANOPOULOS *et al*., 1996).

O fator de consciência, também conhecido como frequência relativa de vitórias, é um parâmetro que atua diminuindo a taxa de vitórias do neurônio que vence com maior frequência, oferecendo oportunidade aos neurônios não vencedores. Assim, todos os neurônios têm chance de ser atualizados no processo de treinamento. O neurônio que mais venceu no passado é punido através da multiplicação do número de vezes que venceu no passado pela distorção (distância do neurônio para o dado de entrada), de modo a desencorajar os neurônios vencedores a atrair novas entradas (BUDURA, 2006).

5.6.3. Rival-Penalize Competitive Learning (RPCL)

Apesar do FSCL formar *clusters* com sucesso sem as unidades mortas, o algoritmo

apresenta o mesmo problema do K-*means*. Torna-se necessário saber previamente o número exato de *clusters*, ou seja, a performance do algoritmo decresce rapidamente se *k* não for bem especificado (AGGARWAL *et al.*, 2003).

Xu e colaboradores (1993) propuseram uma estratégia de determinação automática do número de nós (nodos, neurônios) baseada na determinação do número de nós como função do grau de similaridade entre eles. Esse algoritmo então ficou conhecido como RPCL, que é uma modificação do FSCL que contorna o problema de conhecer o número de nós (XU *et al*., 1993; NAIR *et al*., 2003).

O RPCL tem a capacidade de determinar automaticamente o número de nós no espaço de entrada necessários à melhor representação das classes, ao determinar o número de nós como uma função do grau de similaridade entre esses neurônios (NAIR *et al*., 2003; AGGARWAL *et al*., 2003).

5.6.4. Self-Organizing Map (SOM)

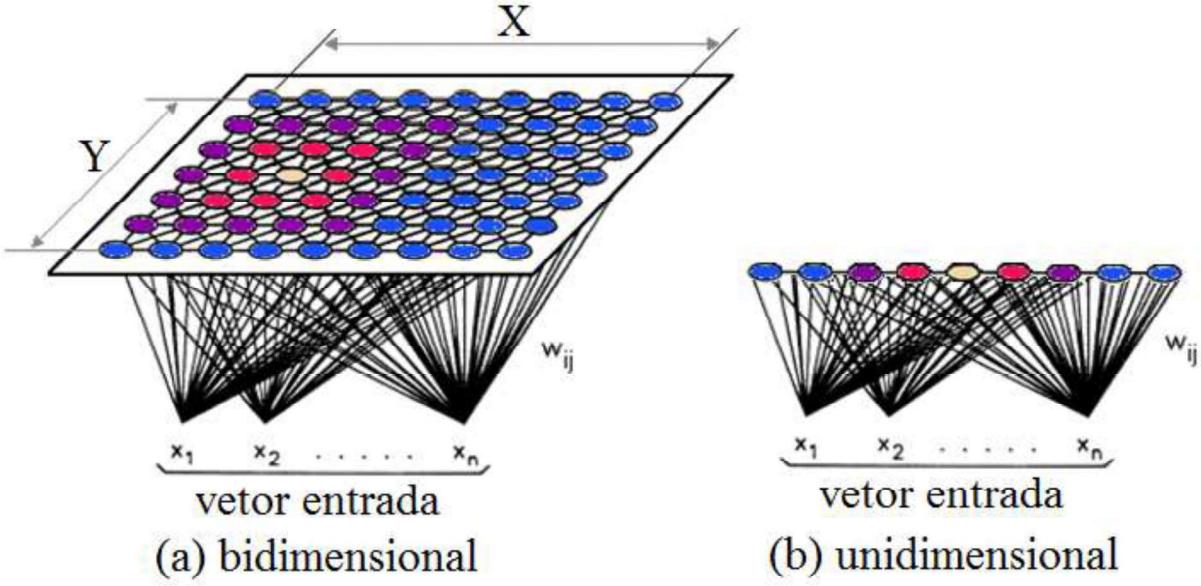
O algoritmo SOM, proposto por Kohonen, é uma das mais bem-sucedidas estratégias de classificação baseada em aprendizado competitivo (KOHONEN, 1982). Diferentemente de estratégias puramente competitivas, como WTA, FSCL e RPCL, o algoritmo SOM possui uma técnica de aprendizado competitivo-cooperativo, no qual neurônios vizinhos competem entre si através de ligações laterais mútuas, e o neurônio BMU determina a localização espacial de uma vizinhança topológica de neurônios excitados que ajustam seus pesos sinápticos de maneira proporcional ao neurônio vencedor (KOHONEN, 1982; RITTER; KOHONEN, 1989; RUMELHART; ZIPSER, 1986).

Em uma RNA SOM, os neurônios são organizados em estruturas de uma dimensão (camada ou *array*) ou mais frequentemente em duas dimensões (grade ou *grid*). Cada neurônio em uma grade está associado a um peso sináptico que tem mesma dimensão do vetor de entrada **x**, e que será apresentado à grade toda. Cabe à RNA SOM aprender esses modelos para poder mapear de forma mais fiel possível em um espaço de saída unidimensional (1 D) ou bidimensional (2 D), os dados multidimensionais e complexos apresentados no espaço de entrada, alocando os dados similares em espaços fisicamente próximos no mapa, como é apresentado na Figura 5.7 (AGGARWAL *et al*., 2003; HAYKIN, 2007; KOHONEN, 1982; KOHONEN, 1990; RUMELHART; ZIPSER, 1986; RITTER; KOHONEN, 1989; XU *et al*., 1993; KOHONEN, 1997).

Para o treinamento da RNA SOM, na fase de competição, a distância Euclidiana de um vetor de entrada para cada peso sináptico é calculada a cada iteração, e o neurônio com a menor distância entre o seu peso e o vetor de entrada é declarado vencedor (Figura 5.7).

Na fase cooperativa, o BMU determinará a posição dos neurônios vizinhos que estiverem excitados, aproximando-os de um segundo raio de abrangência, *R*. Dependendo o valor do raio de abrangência, menos ou mais neurônios podem ser atualizados. Uma função gaussiana atualiza os pesos sinápticos que foram aproximados ao BMU de acordo com *R*, proporcionando uma taxa de decaimento, que se intensifica para os neurônios mais afastados do BMU, mas que devem ser contemplados desde que pertençam ao conjunto vizinhança (ARCOVERDE *et al*., 2011; DA SILVA *et al*., 2010; HAYKIN, 2007).

Figura 5.7 - Mapa topológico (SOM) bidimensional e unidimensional mostrando o neurônio vencedor (BMU) e influência cooperativa sobre vizinhanças.



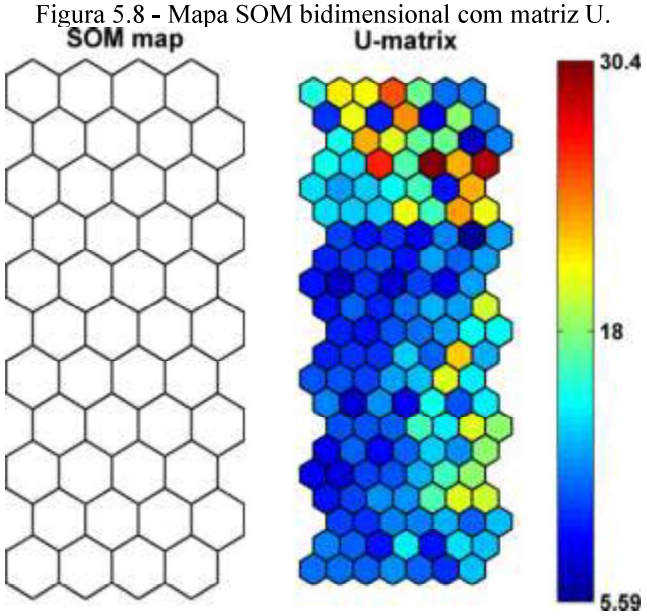
Fonte: adaptado de MOBILIS, 2018.

O processo de treinamento pode ser dividido em duas fases: a primeira, conhecida como ordenação, e uma posterior, a de convergência. Na primeira fase, os raios de influência e taxas de aprendizagem são maiores, para na segunda fase esses valores diminuírem, sendo atualizados. Nos mapas auto organizáveis, as saídas são desconhecidas, fazendo com que o uso de ferramentas estatísticas e o conhecimento de especialistas seja de extrema importância. Após a fase de treinamento, é gerado um mapa topológico representativo do espaço de dados, conhecido como *mapa de contexto*, que deve estar particionado em regiões que definem as classes (*clusters*) a que pertencem esses dados, como apresentado na Figura 5.8. O mapa de contexto é gerado a partir das análises efetuadas após a fase de treinamento e é uma ferramenta que facilita a análise do fenômeno em foco.

Entre as heurísticas adotadas para geração de mapas de contexto, uma das mais importantes é a matriz de distâncias unificadas, matriz-U, que permite analisar visualmente a existência de possíveis agrupamentos. Em uma matriz-U bidimensional (Figura 5.8) os

neurônios geralmente são representados como estruturas hexagonais e as distâncias Euclidianas entre vetores de neurônios são mostradas em imagens em escala de cinza ou, eventualmente, coloridas, de acordo com a distância entre cada neurônio vizinho: quanto mais escuro, maior a distância entre nós, e vice-versa (ULTSCH, 2003; THOMASSEY; HAPPIETTE, 2007; ARCOVERDE *et al*., 2011).

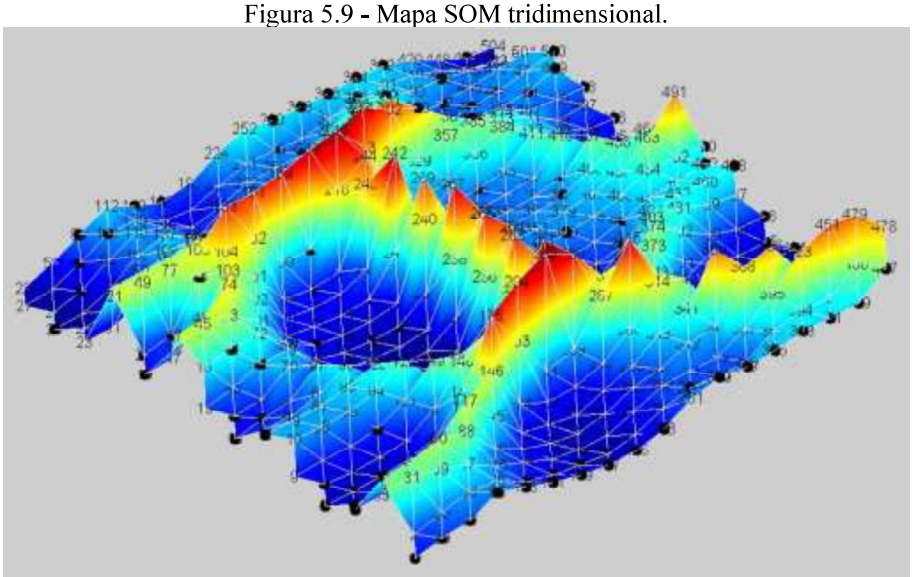
Figura 5.8 - Mapa SOM bidimensional com matriz U.



Fonte: BIEROZA *et al*., 2012.

Em uma matriz-U tridimensional, a distância entre neurônios é mostrada graficamente, com valores de altura que formam estruturas como picos tridimensionais, criando uma paisagem em três dimensões (3 D) no gráfico (Figura 5.9).

Figura 5.9 - Mapa SOM tridimensional.

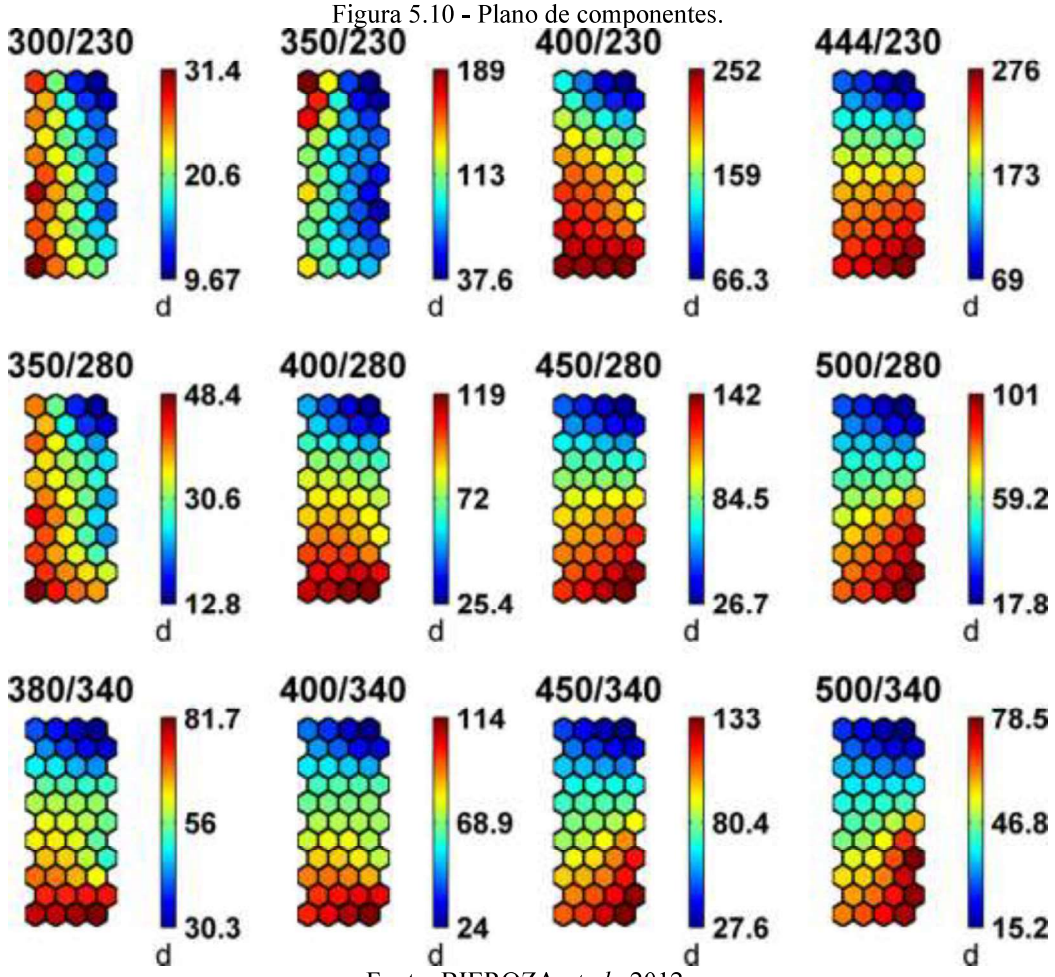


Fonte: PALAMARA *et al*., 2011.

A altura é calculada através da soma das distâncias de todas as vizinhanças imediata normalizada pela maior altura encontrada. O valor, portanto, será mais alto onde há poucos ou nenhum dado, de modo que são formadas "cadeias de montanhas" como fronteiras de *clusters*. De outra forma, a soma será baixa em áreas de alta densidade de dados, de modo que os *clusters* são mostrados como "vales" na paisagem (ULTSCH, 2003; THOMASSEY; HAPPIETTE, 2007).

Outro tipo bastante útil de mapa de contexto para descoberta de características em grupos de dados são os planos de componentes (Figura 5.10). Os planos de componentes permitem a visualização de correlações entre os diversos atributos (variáveis) de entrada de um mapa de Kohonen. São gerados planos de componentes para cada atributo de entrada, e esses planos de componentes são representações gráficas codificadas por cores dos valores de cada variável de vetor de peso, em que os neurônios com cores similares representam características similares (ARCOVERDE *et al*., 2011).

Figura 5.10 - Plano de componentes.



Fonte: BIEROZA *et al*., 2012.

Após o treinamento da RNA SOM, quando é apresentada a ela uma nova amostra como dado de entrada a ser classificada, basta a rede determinar qual foi o neurônio vencedor na

competição, e com a ajuda do mapa de contexto (matriz-U e/ou planos de componentes, por exemplo), identificar qual a classe esse neurônio representa, classificando a nova amostra como pertencente à classe do neurônio vencedor.

O algoritmo padrão de treinamento da RNA SOM é apresentado abaixo (KASKI, 1997).

Algoritmo 4.1 - Sumário do algoritmo SOM.

⟨1⟩ Atribuir um valor inicial aos pesos $w^j \in R^n$, definir os parâmetros de vizinhança e de aprendizagem;
⟨2⟩ Enquanto a condição de parada é falsa, faça:
  ⟨2.1⟩ Para cada j calcule:
    ⟨2.1.1⟩ $D(j) = \underset{j}{\text{argmin}}\{\|w_j - x_j\|\}$
    ⟨2.1.2⟩ Encontre o índice J tal que $D(j)$ seja um mínimo.
    ⟨2.1.3⟩ $\forall\, j \in N_c$ de J, e $\forall\, i$:
      $w_{ij}(novo) = w_{ij}(antigo) + \alpha[x_i - w_{ij}(antigo)]$
  ⟨2.2⟩ Atualize a taxa de aprenizagem
  ⟨2.3⟩ Reduza o raio de vizinhança

em que N_c é o raio de vizinhança.

A taxa de aprendizagem da rede diminui lentamente com o tempo. Existem duas fases na formação de um mapa auto organizável:

- Formação inicial da ordem correta;
- Convergência final.

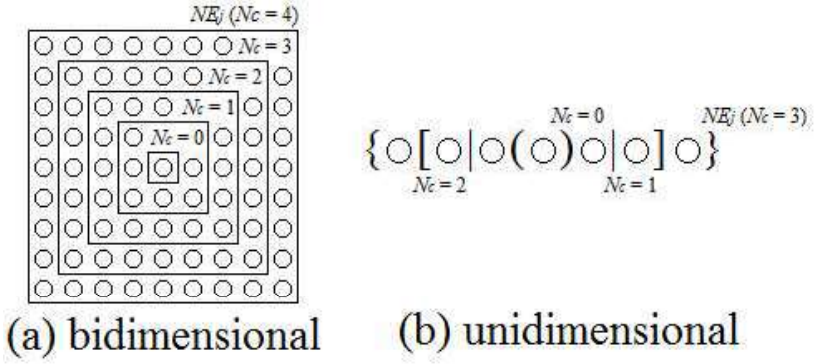
O algoritmo de treinamento começa lendo os vetores dos padrões de entrada x_1, x_2, ..., x_N. Considera-se que são utilizados N padrões diferentes durante o processo de treinamento. O próximo passo é selecionar os valores iniciais para os pesos das conexões w_{ij} (i = 1, ..., n e j = 1, ..., m) e o raio de vizinhança N_c. Kohonen (1982) sugere que os pesos iniciais das conexões sejam valores aleatórios pequenos.

O raio de vizinhança (N_c) tem grande importância na atualização dos pesos das conexões w_{ij}. Os vizinhos de um nó de saída em uma rede bidimensional são definidos dentro de um bloco quadrado, com o nó j sendo o centro do bloco. No caso dos nós de saída em uma rede unidimensional, o nó j é o centro do vetor de saída (Figura 5.11)

No processo de treinamento, N_c é diminuído em 1 após um determinado número de iterações até que N_c seja igual a zero. Neste momento, diz-se que ocorreu uma iteração quando o vetor de padrões x_1, x_2, ..., x_N tiver sido apresentado uma vez. Depois de especificado os pesos iniciais das conexões, o valor de ativação de cada nó na saída pode ser calculado. O nó J com o máximo valor de ativação é selecionado como unidade vencedora. Em seguida, os pesos das conexões para o nó J e todos os nós em sua vizinhança definidos por NE_j são atualizados, como

apresentados na Figura 5.11. Os procedimentos acima são repetidos para cada padrão de entrada. Quando todas as *N* amostras de entrada tiverem sido apresentadas, diz-se que uma iteração está completa.

Figura 5.11 - Vizinhos do nó central *j*, NE_j (N_c).



5.6.5. Neural Gas (NG)

A performance de uma RNA SOM na classificação e principalmente na quantificação de vetores depende de um conhecimento anterior sobre a estrutura topológica do espaço topólogico de entrada (*Manifold*, *M*), que muitas vezes não está disponível ou pode ser de difícil obtenção se a estrutura topológica de *M* for muito heterogênea (por exemplo, quando *M* é composto por subconjuntos de diferentes dimensões ou divisões, e altamente "quebrados"). A rede *Neural Gas* é uma rede desenvolvida a partir do algoritmo SOM, mas capaz de contornar limitações associadas a rigidez dos Mapas Auto Organizáveis: a RNA NG é capaz de quantificar espaços topológicos de entrada heterogeneamente estruturados e aprender relações entre sinais de entrada sem a necessidade de uma pré especificação da topologia da rede (MARTINETZ; SCHULTEN, 1991).

Em uma RNA NG, os pesos sinápticos w^j são ajustados independentemente de qualquer arranjo topológico dos neurônios na rede. Estes ajustes são afetados, na verdade, pelo arranjo topológico dos campos receptivos dentro do espaço de entrada. Em outras palavras, as variações nos pesos sinápticos (Δw^j) não são determinadas pelo arranjo dos neurônios dentro de um retículo pré estruturado, mas pelas distâncias relativas entre as unidades neuronais dentro do espaço de entrada. A mudança mais importante feita no algoritmo NG em relação ao SOM é na regra de adaptação: ao invés da regra "vencedor leva tudo", adotada pelo algoritmo SOM, a regra de ajuste da RNA NG é o "vencedor leva mais" (*winner-take-most*, WTM). A regra WTM consiste no fornecimento das informações sobre o campo receptivo como um conjunto de distorções D_x, associado a cada sinal de entrada **x** (Equação (5.9)). Cada vez que um sinal de entrada **x** é apresentado, é a organização do conjunto D_x em relação a **x** que determina o ajuste dos pesos sinápticos. Os campos receptivos M_i e M_j de neurônios adjacentes *i* e *j* em *M*

desenvolvem conexões representadas pela matriz C_{ij} normalizadas entre 0 e 1. Ao final do aprendizado, é a conectividade da matriz que representa a similaridade (relação de vizinhança) entre os dados de entrada (MARTINETZ; SCHULTEN, 1991).

$$D_x = \{\|v - w^j\|, j = 1, \dots, N\} \tag{5.9}$$

O Algoritmo 4.1 sumariza as instruções de uma RNA NG. Infere-se que cada unidade neural tem que descobrir apenas o quanto e não quais unidades neuronais estão mais proximamente ajustadas ao sinal de entrada.

Algoritmo 4.2 - Sumário do algoritmo NG.

⟨1⟩ Atribuir um valor inicial aos pesos $w^j \in \mathbb{R}^n$ e colocar todos os C_{ij} em zero;
⟨2⟩ Selecionar um vetor de entrada v do espaço de entrada M;
⟨3⟩ Executar uma etapa de adaptação para os pesos de acordo com:

$$w_i(novo) = w_i(velho) + \varepsilon . e^{\frac{k_i}{\lambda_v - w_i(velho)}}, i = 1, \dots, N;$$

⟨4⟩ Se $C_{i0i1} = 0$, colocar $C_{i0i1} = 1$ e $t_{i0i1} = 0$.
Se $C_{i0i1} = 1$ colocar $t_{i0i1} = 0$;
⟨5⟩ Aumentar a idade de todas as conexões de i_0 colocando $t_{i0j} = t_{i0j} + 1$, para todo j com $C_{i0j} = 1$;
⟨6⟩ Remover todas as conexões i_0 que excedam seu tempo de vida, colocando $C_{i0j} = 0$ para todo j com $C_{i0j} = 1$ e $t_{i0j} > T$.
Continuar com 1.

Fonte: DA SILVA *et al.*, 2010.

5.7. Identificação de Processos

A identificação de processos envolve a construção de modelos de processos com dados de entrada/saída obtidos experimentalmente, sem recorrer a quaisquer pressupostos teóricos relativos à natureza fundamental e às propriedades do sistema. Esta atividade é também chamada de identificação de processos (LJUNG, 1996).

Ogunnaike e Ray (1994) afirmaram que a identificação de processos trata os processos como "caixa preta", não se requer conhecimento preciso e completo do processo, somente que os dados de saída sejam obtidos em resposta a mudanças nas entradas. Os modelos obtidos são particularmente úteis para processos pouco entendidos ou processos complexos, se precisam de métodos especiais para obter modelos não lineares.

O processo de construir um modelo empírico por identificação de processos consiste nas seguintes etapas:

1. Definição do problema
2. Coleta e pré-tratamento dos dados
3. Seleção da estrutura do modelo

4. Estimativa dos parâmetros da estrutura selecionada
5. Validação do modelo.

5.8. Modelos Empíricos com Redes Neuronais Artificiais

Para a obtenção de um modelo através de redes neuronais artificiais é necessário o conhecimento prévio dos dados de entrada/saída, para de esta forma aplicar uma estrutura de rede que identificará as relações existentes entre as variáveis independentes e dependentes desse processo. Para que a rede identifique as relações é necessário treiná-la com as informações dessas variáveis. Não é necessário para a modelagem empírica através de uma rede neuronal o conhecimento da fenomenologia do processo.

As redes neuronais são uma poderosa ferramenta para modelagem de sistemas não lineares devido à sua propriedade de generalização universal, ou seja, devido a sua capacidade de se adaptarem aos conjuntos de dados mais diversos (NASCIMENTO e YONEYAMA, 2008). As redes neuronais mais utilizadas em identificação de sistemas não lineares são as do tipo *feedforward* (CLAUMANN, 2003; RANKOVIC; NIKOLIC, 2008).

O *perceptron* multicamada é uma rede neuronal *feedforward* cuja forma compacta para o caso *Multiple Inputs Single Output* (MISO), é dada pela Equação (5.10):

$$F(\mathbf{x},\mathbf{w}) = \psi\left(\sum_{k}\mathbf{w}_{ok}\varphi\left(\sum_{j}\mathbf{w}_{kj}\phi\left(\cdots f\left(\sum_{i}\mathbf{w}_{li}\mathbf{x}_{i}\right)\right)\right)\right) \tag{5.10}$$

em que $f(.)$, $\phi(.)$, $\varphi(.)$, $\psi(.)$; são funções de ativação, $\mathbf{w}$ são vetores de pesos sinápticos de conexões entre neurônios de camadas próximas, e $\mathbf{x}_i$ é o i-ésimo elemento do vetor de entrada $\mathbf{x}$.

O esquema de funções não lineares desse aproximador universal, descrito pela Equação (5.10), não é usual na teoria clássica da aproximação de funções (HAYKIN, 2007).

A modelagem de processos baseada em redes neuronais como identificação de sistemas pode ser descrita da seguinte forma. Um processo pode ser considerado como um sistema de múltiplas entradas e múltiplas saídas (*Multiple Input-Multiple Output*, MIMO) com vetor de entrada $\mathbf{x}$ e vetor de saída $\mathbf{y}$. Considere que o sistema tem um mapeamento de entrada-saída não linear descrito pela relação funcional apresentada na Equação (5.11):

$$\mathbf{y} = f(\mathbf{x}) \tag{5.11}$$

em que os valores das entradas e das saídas são conhecidos e a função f é desconhecida.

A partir dos valores de entradas e saídas se pode montar um conjunto de exemplos rotulados de acordo com a Equação (5.12):

$$E = \{(\mathbf{x}_i, \mathbf{y}_i)\}_{i=1}^{N} \tag{5.12}$$

O conjunto de exemplos pode ser utilizado para treinar uma rede neuronal como um modelo do sistema. Assim, $\mathbf{y}_i$ é a saída desejada para o vetor de entrada $\mathbf{x}_i$. Considere que $\hat{\mathbf{y}}_i$ represente a saída da rede em resposta ao vetor de entrada $\mathbf{x}_i$, de acordo com a Equação (5.13):

$$\hat{\mathbf{y}}_i = f_{NN}(\mathbf{x}_i, \mathbf{w}_i) \tag{5.13}$$

A diferença entre $\mathbf{y}_i$ e $\hat{\mathbf{y}}_i$ fornece um sinal de erro ε_i, que é utilizado para ajustar os parâmetros livres da rede $\mathbf{w}_i$, de forma a minimizar a diferença entre a saída da rede $\hat{\mathbf{y}}_i$ e a saída do sistema desconhecido $\mathbf{y}_i$, calculada sobre o conjunto de treinamento inteiro.

A rede neuronal identifica o sistema desconhecido quando (Equação (5.14)):

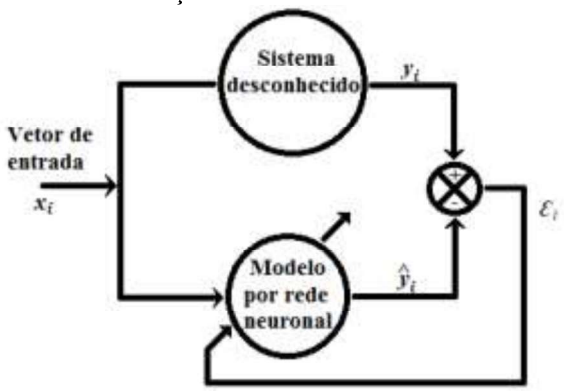
$$\|f_{NN}(\mathbf{x}_i, \mathbf{w}_i) - f(\mathbf{x})\| < \varepsilon \text{ para todo } \mathbf{x} \tag{5.14}$$

em que o erro aproximado ε é um número positivo suficientemente pequeno para a tarefa. A Figura 5.12 mostra o processo de identificação de sistemas utilizando redes neuronais (MORENO-ARMENDARIZ *et al*., 2001; GIL; PÁEZ, 2007).

Para o desenvolvimento de um modelo baseado em redes neuronais são importantes as seguintes etapas:

1. Definição do problema;
2. Coleta dos dados de treinamento e validação;
3. Pré e pós-processamento dos dados;
4. Definição da topologia da rede;
5. Treinamento;
6. Teste e validação;

Figura 5.12 - Identificação de sistemas utilizando redes neuronais.



Definição do problema

Antes de iniciar as tarefas da modelagem é preciso objetivar os propósitos a conseguir com o modelo de rede neuronal. Para efeito de utilização na detecção de falhas, é necessária a obtenção de um modelo complexo o bastante para conseguir agregar as informações da dinâmica do processo, com variáveis de entradas que influenciam as saídas escolhidas, tendo a certeza que existam dados empíricos o bastante para realizar o treinamento e a validação do modelo da rede neuronal. O tempo disponível para a construção do modelo é um fator a se considerar, uma vez que as redes neuronais podem levar longos tempos de treinamento.

Coleta dos dados de treinamento e validação

Após o pré-processamento dos dados e a verificação da existência ou não de inconsistências nos dados operacionais devido a medições incorretas, mau funcionamento de equipamentos do processo ou outros casos, a totalidade dos dados selecionados deve ser distribuída em grupos que serão utilizados para o treinamento e validação do modelo.

Diversos procedimentos de distribuição de dados foram apresentados na literatura, úteis para aplicar a técnica de validação cruzada, que serão explicados mais à frente.

a) *Divisão dos dados em dois conjuntos*

Um conjunto de treinamento utilizado para treinar a rede neuronal e um conjunto de teste utilizado para validar o modelo.

b) *Divisão dos dados em três conjuntos*

Conjuntos de treinamento, validação e teste (HAYKIN, 2007).

c) *Divisão de dados exaustiva (Exhaustive Data Splitting; ARLOT, 2010)*

Divide-se os dados utilizando todas ou algumas das $\binom{n}{n_v}$ combinações que podem ser obtidas quando o total de dados se divide em dois grupos.

Em que n são os dados disponíveis (pares entradas-saída), n_v é o conjunto de pares de entrada-saída utilizado para a validação do modelo e $n_t = n - n_v$ é o conjunto de pares entrada-saída utilizado para treinar a rede neuronal e determinar seus parâmetros.

d) Divisão de dados parcial (Partial Data Splitting; ARLOT, 2010)

Considerando que $\binom{n}{n_v}$ conjuntos de treinamento podem ser computacionalmente intratáveis, vários esquemas de divisão parcial de dados foram propostas como alternativas.

Pré e pós-processamento de dados

Com o objetivo de eliminar a interferência das diferentes magnitudes das variáveis no processo de aprendizagem da rede, todos os dados são pré-processados (normalizados), de maneira que os valores das entradas e das saídas se encontrem no mesmo intervalo. Após a validação do modelo, os valores são pós-processados para retorná-los às grandezas originais.

Os procedimentos de pré e pós-processamento dos dados são diversos e os autores devem utilizá-los de acordo com o problema em questão, por exemplo, o intervalo de 0 a 1 para entradas e saídas pode ser usado por compatibilidade com a função de ativação sigmoide. Outros autores julgam que como os valores de 0 a 1 são valores infinitos, para a função sigmoide é recomendável diminuir o intervalo de valores para o intervalo [0,1;0,9] ou ainda a [0,2;0,8], para facilitar a convergência durante o treinamento da rede.

A expressão linear geral para adimensionalizar ou normalizar as entradas de um vetor $\mathbf{x}$, em um mesmo intervalo dado por $[\mathbf{L}_{inf}, \mathbf{L}_{sup}]$ é apresentada na Equação (5.15).

$$\mathbf{x}_n(i) = \left[\frac{\mathbf{x}_{máx} - \mathbf{x}(i)}{\mathbf{x}_{máx} - \mathbf{x}_{min}}\right]\mathbf{L}_{inf} + \left[\frac{\mathbf{x}(i) - \mathbf{x}_{min}}{\mathbf{x}_{máx} - \mathbf{x}_{min}}\right]\mathbf{L}_{sup} \tag{5.15}$$

A expressão linear geral para retornar o vetor de saídas normalizadas $\mathbf{y}_n$ do modelo, após validação, às grandeza originais é apresentada na Equação (5.16).

$$\mathbf{y}(i) = \left[\frac{\mathbf{L}_{sup} - \mathbf{y}_n(i)}{\mathbf{L}_{sup} - \mathbf{L}_{inf}}\right]\mathbf{y}_{min} + \left[\frac{\mathbf{y}_n(i) - \mathbf{L}_{inf}}{\mathbf{L}_{sup} - \mathbf{L}_{inf}}\right]\mathbf{y}_{máx} \tag{5.16}$$

A adimensionalização logarítmica é obtida utilizando as seguintes Equações (5.17) e (5.18).

$$\mathbf{y}_n(i) = \frac{\log_{10}(\mathbf{y}(i) - a)}{b} \tag{5.17}$$

em que:

$$a = \frac{\mathbf{L}_{sup}[\log_{10}(\mathbf{y}_{min})] - \mathbf{L}_{inf}[\log_{10}(\mathbf{y}_{máx})]}{\mathbf{L}_{sup} - \mathbf{L}_{inf}}$$
$$b = \frac{\log_{10}(\mathbf{y}_{máx}/\mathbf{y}_{min})}{\mathbf{L}_{sup} - \mathbf{L}_{inf}} \tag{5.18}$$

A adimensionalização logarítmica se torna melhor que a linear quando no conjunto de dados de entrada o maior valor é muito superior ao menor valor (SILVA, 1998).

Definição de topologia da rede neuronal

Uma das questões mais importantes ainda não resolvidas na literatura sobre redes neuronais é qual topologia deve ser utilizada para um problema dado. A seleção da topologia requer que seja escolhido o número apropriado de neurônios distribuídos em camadas e as suas conexões (ANDERS; KORN, 1999). No problema de aproximação de funções uma regra prática para obter uma boa generalização é usar a rede neuronal de menor dimensão que possa ajustar os dados (REED, 1993), pois uma rede neuronal com tamanho mínimo é menos provável que aprenda também o ruído dos dados de treinamento (em problemas reais), e assim pode generalizar melhor com novos dados não considerados no treinamento da rede.

Durante o processo de determinação do melhor modelo de rede neuronal são propostas várias topologias de rede, e, então, utilizando diversos procedimentos e critérios, é selecionada a topologia ótima para o modelo de rede neuronal representativo do processo.

A topologia do modelo de rede neuronal pode ser obtida por aplicação dos seguintes métodos:

- Parada antecipada.
- Validação cruzada.
- Procedimentos estatísticos.
- Construção de redes.
- Poda de redes (*pruning*).
- Uso de algoritmos evolutivos.

Os dois primeiros métodos são explicados posteriormente, referido a validação e teste do modelo baseado em redes neuronais, e um detalhamento dos outros métodos pode ser

encontrado na literatura (MOZER; SMOLENSKY, 1989; FALHMAN; LEBIERE, 1990; LECUN *et al.*, 1990; HASSIBI *et al.*, 1993; HAGIWARA, 1994; STAHLBERGER; RIEDMILLER, 1997; ANDERS; KORN, 1999; ARIFOVIC; GENÇAY, 2001; KENNEDY, 2003; ATTIK *et al.*, 2004; HAYKIN, 2007; CONFORTH; MENG, 2008; REITERMANOVA, 2008; ADMUTHE *et al.*, 2009; TAHMASEBI; HEZARKHANI, 2010; AGHBASHLO *et al.*, 2011).

Treinamento da rede neuronal

No procedimento de aprendizagem, um mapeamento de entrada-saída é codificado em pesos sinápticos e limiares de um *perceptron* multicamadas, e a partir dessa perspectiva, o processo de aprendizagem na etapa de treinamento equivale a uma escolha dos parâmetros da rede para este conjunto de dados (HAYKIN, 2007).

Na etapa de treinamento realiza-se com algoritmos de treinamento, taxas de aprendizado, momentum, tempo de treinamento, procurando os melhores parâmetros para a rede neuronal.

A definição da topologia, o treinamento da rede neuronal e a validação do modelo de rede neuronal estão extremamente interligados, um treinamento em excesso para uma determinada topologia leva a uma capacidade pobre do modelo para generalizar com dados não utilizados na fase de treinamento. A tendência natural é que o erro de treinamento diminua quando o número de neurônios ocultos é aumentado ou quando mais iterações de treinamento são utilizadas. Em ambos os casos, pode resultar um problema de supertreinamento ou *overtraining*, pelo qual o ajuste com dados de treinamento torna-se quase perfeito, mas o erro de generalização torna-se pior.

As redes neuronais mais utilizadas em identificação de sistemas não lineares são as do tipo *feedforward* e grande parte desse sucesso pode ser atribuído ao algoritmo de treinamento supervisionado e iterativo conhecido por retropropagação do erro (*Backpropagation error*), discutido por Rumelhart, Hinton e Williams em 1986 (CLAUMANN, 2003; RANKOVIC; NIKOLIC, 2008).

Métodos para melhorar a velocidade de aprendizagem do algoritmo de retropropagação

No contexto das redes multicamada *feedforward* na literatura algumas das desvantagens referidas ao algoritmo de *backpropagation* são:

O algoritmo tem uma restrição computacional, os cálculos realizados pelo neurônio são influenciados apenas por aqueles neurônios que estão em contato físico com ele (HAYKIN, 2007).

Uma das maiores desvantagens do algoritmo de *backpropagation* é a sua lenta velocidade de convergência (CHO; KIM, 1993).

Em muitas aplicações, o número de interconexões ou pesos em uma rede neuronal é tão grande que o tempo de aprendizagem para o algoritmo de retropropagação convencional pode se tornar excessivamente longo (JOHANSSON *et al.*, 1991).

Uma variedade de métodos foi aplicada às redes neuronais para melhorar a velocidade de aprendizagem do algoritmo de *backpropagation*. Lahmiri (2011) identificou duas categorias principais. A primeira categoria usa técnicas heurísticas desenvolvidas com base em uma análise do desempenho do método da descida mais íngreme (*steepest descent*) padrão, nesta categoria estão variedades do algoritmo do gradiente descendente com taxa de aprendizagem adaptativa e/ou momentum. A segunda categoria usa técnicas de otimização numérica padrão, nesta categoria estão incluídos os métodos de gradiente conjugado, o método de Levenberg-Marquardt e os métodos quase-Newton.

Dentre esses métodos, o de Levenberg-Marquardt adapta-se bem a criação de modelos para a detecção de falhas em sistemas de controle, o qual será explicado posteriormente.

O algoritmo Levenberg-Marquardt para o treinamento da rede neuronal

Utilizando-se o algoritmo de retropropagação do erro com algumas modificações é possível utilizar um algoritmo de minimização de mínimos quadrados para o treinamento de redes neuronais, denominado de Levenberg-Marquardt (HAGAN; MENHAJ, 1994).

Quando a função que é minimizada tem a forma da Equação (5.19), o problema de otimização é chamado de minimização de mínimos quadrados não linear (*nonlinear least squares minimization*):

$$\mathbf{I} = f(\mathbf{w}) = \frac{1}{2}\sum_{i=1}^{N} e_i^2(\mathbf{w}) \tag{5.19}$$

O índice de desempenho **I** pode ser definido como a soma do quadrado do erro e_i, associado ao *i*-ésimo padrão de treinamento da rede.

A propriedade distintiva dos problemas de mínimos quadrados é que, dada a matriz Jacobiana, J, a Hessiana, H, é definida pela Equação (5.20).

$$H \approx \nabla^2 f(\mathbf{w}) = J(\mathbf{w})^T J(\mathbf{w}) \tag{5.20}$$

A regra de atualização dos pesos do algoritmo de Levenberg-Marquardt pode ser explicada como o resultado de uma modificação feita no método de Newton da seguinte forma.

Um modo intuitivo simples para encontrar o mínimo de uma função é modificar os parâmetros utilizando o gradiente da função, de acordo com a Equação (5.21).

$$\mathbf{w}_{i+1} = \mathbf{w}_i - \lambda \nabla f(\mathbf{w}_i) \tag{5.21}$$

A atualização dos parâmetros depende do gradiente da função, o gradiente simples tem vários problemas de convergência (RANGANATHAN, 2004). O uso do método de Newton para resolver a equação $\nabla f(\mathbf{w}) = 0$ permite utilizar informação da curvatura, bem como do gradiente. Expandindo-se o gradiente de f usando uma série de Taylor em torno de w_0 tem se a Equação (5.22).

$$\nabla f(\mathbf{w}) = \nabla f(\mathbf{w}_0) + (\mathbf{w} - \mathbf{w}_0)^T \nabla^2 f(\mathbf{w}_0) + \cdots \tag{5.22}$$

Considerando que f é quadrático em torno de w_0 e resolvendo para o mínimo de w, tem-se a regra de atualização para o método de Newton de acordo com a Equação (5.23).

$$\mathbf{w}_{i+1} = \mathbf{w}_i - \left(\nabla^2 f(\mathbf{w}_i)\right)^{-1} \nabla f(\mathbf{w}_i) \tag{5.23}$$

em que $\mathbf{w}_0$ foi substituído por $\mathbf{w}_i$ e $\mathbf{w}$ por $\mathbf{w}_{i+1}$.

O método de Newton usa a suposição de que f é quadrático, nesse caso a Hessiana H não precisa ser avaliada exatamente, pode ser utilizada uma aproximação, dada pela Equação (5.22). Então, o método converge rapidamente, mas a velocidade de convergência é sensível ao ponto inicial (RANGANATHAN, 2004).

Levenberg propôs um algoritmo baseado nessa observação, considerando a Hessiana e a matriz identidade na regra de atualização apresentada na Equação (5.24).

$$\mathbf{w}_{i+1} = \mathbf{w}_i - (H + \lambda I)^{-1} \nabla f(\mathbf{w}_i) \tag{5.24}$$

em que H é a matriz Hessiana avaliada em $\mathbf{w}_i$.

A desvantagem é que, quando o valor de λ é grande, a matriz Hessiana calculada não é utilizada. Uma vez que a Hessiana é proporcional à curvatura, é possível aproveitar melhor a sua contribuição substituindo a matriz identidade com a diagonal da matriz Hessiana, esse discernimento crucial foi fornecido por Marquardt, resultando na regra de atualização de Levenberg-Marquardt, dada pela Equação (5.25).

$$\mathbf{w}_{i+1} = \mathbf{w}_i - (H + \lambda \text{diag}[H])^{-1} \nabla f(\mathbf{w}_i) \quad (5.25)$$

Desta forma, é obtido um grande deslocamento na direção com baixa curvatura e um maior deslocamento na direção com alta curvatura, e o problema clássico de "*Valley error*" não ocorre novamente.

O algoritmo de Levenberg-Marquardt parece ser o método mais rápido para o treinamento de redes neuronais de tamanho moderado (até algumas centenas de parâmetros livres na rede) (FIORIN *et al*., 2011).

Validação e teste do modelo baseado em redes neuronais

A validação cruzada (*cross validation*)

Treinar um modelo com um conjunto de dados e testar o seu desempenho estatístico com os mesmos dados gera um resultado excessivamente otimista. A razão para isto é que no treinamento, os parâmetros do modelo são otimizados para refletir as peculiaridades do conjunto de dados utilizados.

A validação cruzada é uma ferramenta padrão em estatística que fornece procedimentos para obter uma boa estimativa do desempenho estatístico do modelo, testando a sua saída com um conjunto de dados novos não utilizados no treinamento (HAYKIN, 2007).

Variantes da validação cruzada

a) Divisão dos dados em dois grupos

O *hold out method* é o tipo mais simples de validação cruzada, o conjunto de dados é dividido em um subconjunto de treinamento e um subconjunto de validação que é utilizado para testar o desempenho do modelo (ARLOT, 2010). O cálculo é simples, no entanto os resultados podem ser significativamente diferentes dependendo de como foi feita a divisão.

b) Divisão dos dados em três conjuntos

Segundo Haykin (2007) para a validação cruzada, primeiro os dados são aleatoriamente divididos em conjunto de treinamento e conjunto de teste (*test set*), o conjunto de dados de treinamento é ainda dividido em dois subconjuntos: um subconjunto de estimativa usado para selecionar o modelo e um subconjunto de validação usado para validar o modelo com um conjunto de dados diferente do usado para estimar os parâmetros. No entanto, o *overfitting* ainda pode estar

presente no melhor modelo validado. Para evitar o *overfitting*, a capacidade de generalizar do modelo selecionado é medida com o conjunto de teste, o qual é diferente do subconjunto de validação.

c) Divisão de dados exaustiva (exhaustive data splitting)

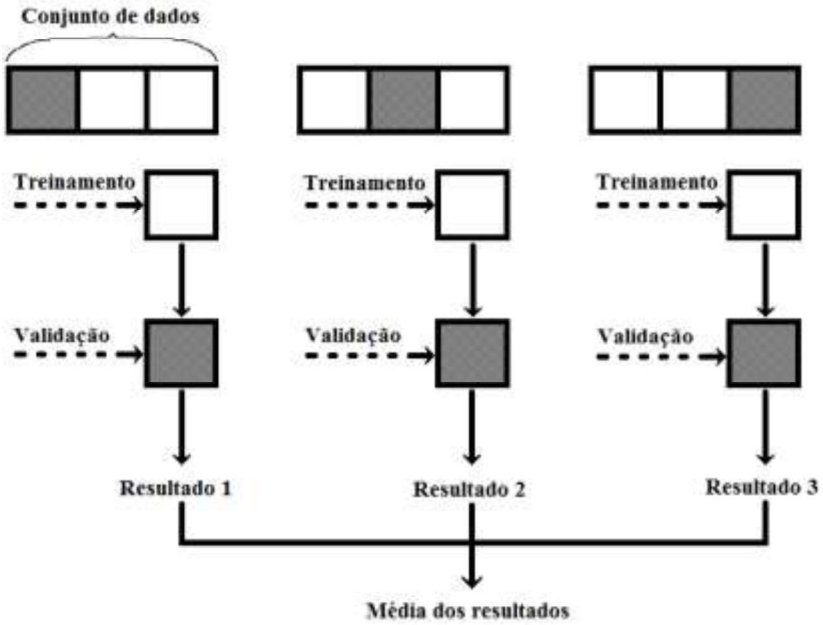
- *Leave-one-out*: Para um conjunto de n integrantes se realizam n experimentos, em cada um deles o conjunto de validação tem um par de entrada-saída e o conjunto de treinamento com $n-1$ pares entrada-saída, que se usam como exemplos de treinamento. O número de conjuntos de treinamento é n (ARLOT, 2010).
- *Leave-p-out*: Realizam-se vários experimentos, o conjunto de n pares entrada-saída é dividido em dois grupos em cada experimento: Conjunto de validação com n_v pares entrada-saída e conjunto de treinamento com $n-n_v$ pares entrada-saída que se usam como exemplos de treinamento (ARLOT, 2010).

  O número de conjuntos de treinamento é (Equação (5.26)):

$$T = \binom{n}{n_v} \tag{5.26}$$

- *Divisão de dados parcial* (*parcial data splitting*): Considerando que $\binom{n}{n_v}$ conjuntos de treinamento podem ser computacionalmente intratáveis, esquemas de divisão parcial de dados foram propostas como alternativas.
- *Validação cruzada V-fold*: Foi introduzida por Geisser (1975) como alternativa ao computacionalmente caro *Leave-one-out*. Dividem-se os n pares de entrada-saída em V subconjuntos de aproximadamente igual cardinalidade n/V, cada subconjunto sucessivamente desempenha o papel de subconjunto de validação cruzada. O conjunto de treinamento tem $n-n/V$ pares de entrada-saída (Figura 5.13).

Figura 5.13 - Método de *V-fold* para validação cruzada, com $V = 3$.



- *Validação cruzada incompleta balanceada* (BICV): Foi proposta por Shao (1993), neste método se aplicam projetos de blocos incompletos balanceados (BIB) para distribuir o total de dados n em grupos de treinamento e validação. O BICV pode ser visto como uma alternativa ao método de validação cruzada *V-fold* quando o tamanho do conjunto de treinamento n_t é pequeno (ARLOT, 2010). Neste caso, o conjunto de validação tem n_v pares de entrada-saída de acordo com a Equação (5.27).

$$n_v = n - n_t \tag{5.27}$$

Para a aplicação do desenho BIB se utilizam os parâmetros a e k:

$a = n$ pares entrada-saída.

$k = n_v$ pares entrada-saída por bloco.

Então, o número de blocos é dado pela Equação (5.28).

$$b = \binom{n}{n_v} \tag{5.28}$$

Atribui-se uma diferente combinação de pares entrada-saída para cada bloco. O número de repetições para cada par entrada-saída é dado pela Equação (5.29).

$$b = \binom{n-1}{n_v-1} \tag{5.29}$$

O número de vezes que cada par entrada-saída ocorre no mesmo bloco é apresentado na Equação (5.30).

$$\lambda = \binom{n-2}{n_v-2} = \frac{r(k-1)}{a-1} \tag{5.30}$$

O parâmetro λ deve ser um número inteiro (MONTGOMERY, 2001).

No BIB, dois pares entrada-saída ocorrem juntos o mesmo número de vezes em relação à os outros pares.

d) Aprendizagem-teste repetido (Repeated learning-testing)

Foi introduzida por Breiman e colaboradores (1984). Neste método os n pares entrada-saída são divididos repetidas vezes aleatoriamente em um conjunto de treinamento n_t de tamanho $n(1-p)$ e um conjunto de teste n_v de tamanho np, em que $0 < p < 1$, $n = n_t + n_v$, tipicamente $n_t \geq n_v$ (BURMAN, 1989). Se $p = 1/3$ cada conjunto de teste é de tamanho $n/3$ e cada conjunto de treinamento é de tamanho $2n/3$. Para cada divisão são obtidas estimativas baseadas nos dados do conjunto de treinamento que depois são testadas com os dados do conjunto de teste.

A validação cruzada para seleção de modelos de redes neuronais

A validação cruzada é uma importante ferramenta para seleção de modelos, a escolha do melhor modelo de rede neuronal é baseada na sua capacidade de generalização com dados novos contidos no conjunto utilizado para testar o desempenho do modelo.

Segundo Raghavarao e Padgett (2005), a seleção do melhor modelo está baseada no cálculo da média ao quadrado dos erros de previsão, apresentado na Equação (5.31).

$$\overline{E} = \frac{(Y-\hat{Y})'(Y-\hat{Y})}{n_v} \tag{5.31}$$

Em que: **Y** é um vetor das n_v respostas que correspondem ao conjunto de validação. $\hat{\mathbf{Y}}$ é o vetor de respostas estimadas para o conjunto de validação baseadas no modelo obtido com os n_t pares entrada-saída utilizados para treinar a rede e determinar seus parâmetros. $\overline{E}$ é calculado para todas ou uma parte das $\binom{n}{n_v}$ combinações que podem ser obtidas quando o total

de dados se divide em dois grupos. O modelo selecionado possui o menor valor na soma das médias ao quadrado dos erros de previsão resultantes.

Quando se utiliza a distribuição de dados em três conjuntos segundo o esquema sugerido por Haykin (2007), utiliza-se a Equação (5.32).

$$n = (n_e + n_v)_t + n_{tt} \tag{5.32}$$

em que n_{tt} é o número de pares entrada-saída do conjunto de teste; $(n_e + n_v)_t$ é o número de pares entrada-saída do conjunto de treinamento divididos em n_e pares entrada-saída para o conjunto de estimativa e n_v pares entrada-saída para o conjunto de validação. Na Equação (5.31), Y é um vetor das n_{tt} respostas que correspondem ao conjunto de teste.

A validação cruzada e parada antecipada (*early stopping*)

A parada antecipada é uma técnica utilizada para evitar o *overtraining* na qual se determina o ponto de parada do treinamento por observação do erro produzido em predições com integrantes do grupo de validação cruzada.

O procedimento mais simples para aplicar o *early stopping* é dividir o conjunto de dados em dois subconjuntos, um deles para treinamento e o outro para validação cruzada. O subconjunto de treinamento é utilizado para modificar os pesos, e o subconjunto de validação é utilizado para estimar a capacidade de generalização do modelo (REED, 1993). Quando o treinamento progride, o erro deve diminuir tanto no treinamento como no conjunto de validação. Com maior treinamento, passando a existir o aumento do erro no subconjunto de validação ao invés da diminuição é uma indicação da existência de *overtraining*, portanto, o ponto em que é observado o erro mínimo é o ponto de parada do treinamento. Segundo Haykin (2007), o que a rede aprende além do ponto de erro mínimo é essencialmente ruído contido no conjunto de treinamento.

No entanto, a situação real é muito mais complexa. As curvas de generalização reais obtidas com o subconjunto de validação quase sempre têm mais de um mínimo local. Prechelt (1998a) reporta um caso com 16 mínimos locais, e afirma que é impossível, em geral, dizer a partir do início da curva se o mínimo global já foi visto ou não, ou seja, se um aumento no erro de generalização indica *overfitting* real ou é apenas intermitente.

Portanto, a aplicação do *early stopping* nos casos de mínimos múltiplos leva a um gasto de treinamento significativo para garantir bons resultados em relação à capacidade de generalização do modelo. Nestes casos com o uso de só dois subconjuntos: treinamento e validação; existe o risco de que ainda o *overfitting* possa estar presente no modelo validado, se

o treinamento foi parado em um mínimo local e não no mínimo global. Porém, a utilização do método de *early stopping* foi realizada neste trabalho uma vez que o objetivo global era o estudo da metodologia de detecção de falhas propriamente dita, e não um estudo profundo de redes neuronais, sendo que o *early stopping* cumpre o desejado.

A divisão dos dados em três subconjuntos: treinamento, validação e teste; para aplicar *early stopping* é outra possibilidade mencionada na literatura. Prechelt (1998b) utilizou o seguinte procedimento para medir o desempenho da rede: divisão do conjunto de dados em duas partes distintas, dados de treinamento e dados de teste.

Os dados de treinamento foram subdivididos em um conjunto de exemplos de treinamento utilizados para ajustar os pesos da rede e um conjunto de exemplos de validação utilizados para estimar o desempenho da rede durante o treinamento, conforme exigido pelos critérios de parada antecipada.

Os dados de teste foram usados para estimar o desempenho da rede após a conclusão do treinamento.

Segundo Kearns (1996) com dois grupos: treinamento e validação cruzada, uma porcentagem de aproximadamente 10% do conjunto total de dados é adequada para o conjunto de validação cruzada. Haykin (2005) utiliza como referência estes resultados e indica que uma escolha apropriada é distribuir 20% do conjunto de treinamento para o subconjunto de validação. Nesse caso, o conjunto de treinamento é dividido em subconjunto de estimativa e subconjunto de validação, e utiliza-se outro conjunto de teste diferente desse de validação para medir a capacidade de generalização do modelo.

Bose e Liang (1996) assinalam que 10% do conjunto de treinamento pode ser utilizado como subconjunto de validação. Yahya e colaboradores (2010) utilizaram *early stopping* com três subconjuntos: subconjunto de treinamento (60% dos dados), subconjunto de validação (20% dos dados) e subconjunto de teste (20% dos dados).

5.9. Detecção de Falhas

As RNAs podem realizar a detecção de falhas em sistemas de controle detectando as condições normais e anormais dos parâmetros dados, que levam a variadas falhas, dependendo do problema em questão. A condição normal representa a situação sem falhas e as condições anormais representam as ocorrências de falhas. O principal objetivo de selecionar as RNAs como ferramenta de detecção é a inaptidão de formar relações matemáticas para o problema devido a não linearidade entre as variáveis de entrada e as variáveis de saída do sistema, a boa generalização das redes neuronais, a rápida operação em tempo real e a habilidade de realizar complicados mapeamentos sem as equações e relações funcionais do problema.

O treinamento de RNAs difere principalmente se esta é supervisionada ou não-supervisionada, ou de acordo com a sua topologia.

A utilização das redes neuronais supervisionadas para detecção de falhas possui duas etapas; treinamento e teste. Durante a etapa de treinamento, a rede neuronal é treinada para capturar as relações fundamentais entre as variáveis de entrada e saída da rede escolhidas. Após o treinamento, a rede neuronal é testada com um conjunto de dados de teste, que não foi utilizado durante o treinamento.

Uma vez que a rede neuronal supervisionada é treinada e testada, ela está pronta para detectar uma falha em diferentes condições de operação. As seguintes questões devem ser abordadas durante o desenvolvimento do modelo para detecção de falhas.

a) Seleção das variáveis de entrada e de saída;
b) Produção dos dados de treinamento;
c) Normalização dos dados;
d) Seleção da estrutura da rede neuronal;
e) Treinamento da rede

Seleção das variáveis de entrada e saída

Para a utilização de aplicações de aprendizagem de máquina supervisionada, é importante selecionar devidamente as variáveis de entrada, uma vez que as RNAs devem aprender a relação entre as variáveis de entrada e as variáveis de saída baseados nos pares entrada-saída fornecidos durante o treinamento.

Nos sistemas de detecção de falhas baseados em redes neuronais supervisionadas, as variáveis de entrada representam o comportamento das variáveis manipuláveis e/ou controladas, enquanto a(s) variável(eis) de saída pode(m) ser a(s) resposta(s) ao(s) comportamentos das variáveis de entrada nas RNAs de regressão; ou podem ser então a probabilidade do respectivo dado de entrada fazer ou não parte de uma determinada classe nas RNAs de classificação.

Produção dos dados de treinamento

A produção dos dados de treinamento é um importante passo no desenvolvimento de modelos de redes neuronais artificiais. Para alcançar um bom desempenho da rede neuronal, os dados de treinamento devem representar uma variedade completa das condições operacionais do equipamento em questão, contendo todas as possíveis ocorrências de falhas conhecidas.

Normalização dos dados

Se os dados produzidos são diretamente alimentados a rede neuronal como padrões de treinamento, as variáveis de entrada com altos valores podem tender a suprimir a influência das variáveis de entrada com valores menores. Também, se os dados brutos são diretamente aplicados a rede, existe o risco dos neurônios simulados atingirem as condições saturadas. Se os neurônios se tornam saturados, então as modificações nos valores da entrada podem produzir uma pequena ou nenhuma modificação no valor da variável de saída. Isto afeta o treinamento da rede em grande parte. Os dados são normalizados antes da rede neuronal ser treinada, de forma que a rede vai dar igual prioridade a todas as entradas.

A normalização dos dados comprime a gama dos dados de treinamento entre 0 e 1 ou -1 a +1 dependendo do tipo da função de ativação.

Seleção da estrutura da rede neuronal

Para fazer com que uma rede neuronal supervisionada realize alguma tarefa específica, é necessário escolher como que as unidades são conectadas umas às outras. Isto inclui a seleção do número de nós ocultos e o tipo de função de ativação utilizada. O número de unidades ocultas está diretamente relacionado às capacidades da rede neuronal. Para o melhor desempenho da rede neuronal, um número ótimo de unidades ocultas deve ser devidamente determinado utilizando o procedimento de tentativa e erro.

O algoritmo de treinamento de retropropagação (*backpropagation*), que propaga o erro da camada de saída para a camada oculta para atualizar a matriz de pesos sinápticos é o mais comumente utilizado para redes neuronais *feedforward*. Os dados de treinamento gerados são normalizados e aplicados a rede neuronal com suas respectivas saídas, para aprender a relação entrada-saída.

5.10. Estudo de Caso - Processo de Produção de Ciclopentanol

Para ilustrar a aplicação da detecção e identificação de falhas através de redes neuronais artificiais supervisionadas de alimentação direta (ANN FF) e de regressão (ANN-R), e não supervisionadas *winner-take-all* (ANN WTA), foi utilizado o processo de produção de Ciclopentanol a partir da reação de van der Vusse, explicada anteriormente na Seção 3.5.

5.10.1. Redes Neuronais Artificiais Supervisionadas de Alimentação Direta

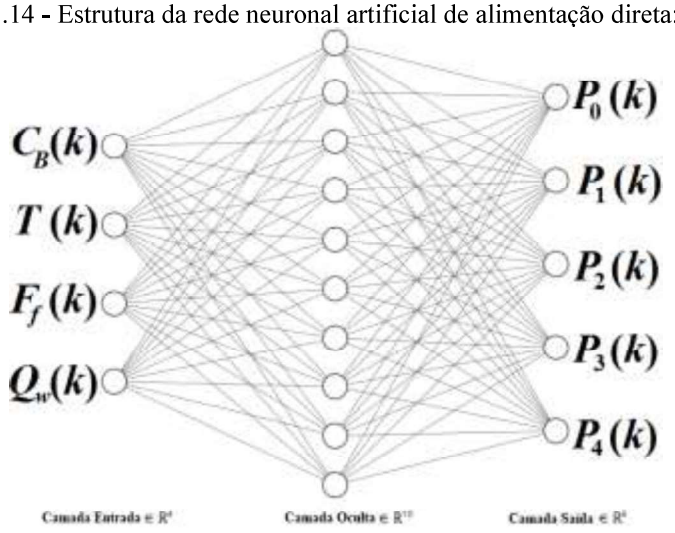
Para as redes neuronais artificiais de alimentação direta de classificação (*Artificial Neural Networks Feedforward*, ANN FF, ou ANN-C), do mesmo modo que para o SVC, foi necessário treinar a rede neuronal com as informações de treinamento de todas as falhas.

controladas, C_B e T, e variáveis manipuladas, F_f e $\dot{Q}_w$, como entradas do modelo ANN de alimentação direta.

Para a saída do modelo ANN FF, foram estabelecidas probabilidades para que cada dado faça parte de cada uma das classes, sendo essas classes as operações faltosas (Pert. C_{Af} - 1, Falha C_B - 2, Falha T - 3 e Emperramento F_f - 4) e a operação normal (0). Para a saída do treinamento e validação, foram estabelecidas juntamente às variáveis de treinamento e validação o vetor de probabilidades $P = [P_0 \quad P_1 \quad P_2 \quad P_3 \quad P_4]^T$, para $P_i = 1$ se i = falha, e $P_j = 0$ se $j \neq$ falha.

A rede então foi definida com a entrada $x = \left[C_B \quad T \quad F_f \quad \dot{Q}_w\right]^T$ e saída $y = [P_0 \quad P_1 \quad P_2 \quad P_3 \quad P_4]^T$. A estrutura da rede foi definida através de estudo empírico (única camada oculta com o dobro de neurônios da camada de saída) com 4 nós na camada de entrada, cada nó para cada uma das variáveis do vetor x, com uma camada interna (oculta) de neurônios com 10 nós, e a camada de saída com 5 nós, cada um dele especificando a probabilidade de determinado dado fazer parte de cada uma das classes (Figura 5.14).

Figura 5.14 - Estrutura da rede neuronal artificial de alimentação direta: 4:10:5.



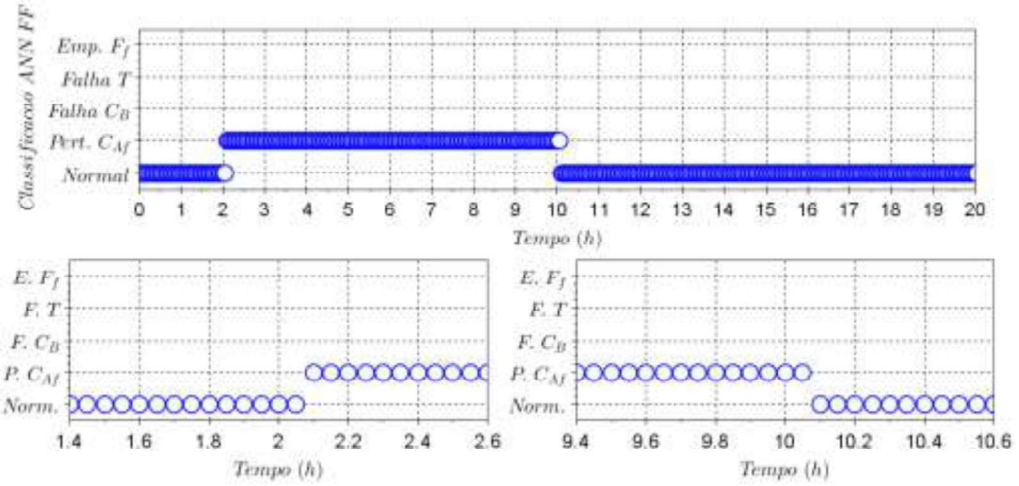
Para o treinamento do sistema de detecção e diagnóstico através de ANN de alimentação direta, foram utilizados dois terços dos dados, enquanto que o um terço restante foi utilizado para validação, de acordo com o capítulo 3. Para cada dois dados selecionados para treinamento, foi selecionado um para validação. Para o treinamento da rede, foram estabelecidas três mil (3000) épocas. Realizando a análise do MSE dos dados de validação do modelo, estabeleceu-se um critério de erro menor que 10^{-5} para definir a quantidade de épocas de treinamento. Cada época no processo de treinamento da rede neuronal é utilizada para melhorar a matriz de pesos **W**, de acordo com a Equação (5.23). Após diversos testes para a seleção da melhor taxa de

aprendizagem, para o determinado estudo de caso a taxa de aprendizagem foi estabelecida como 0,9.

Após o treinamento e validação da rede, foi armazenada a matriz peso **W** e as informações pertinentes à rede ANN de alimentação direta, como a estrutura das camadas da rede e a taxa de aprendizado cuja rede foi treinada, que posteriormente foi utilizada para a classificação de novos dados. Para a obtenção da classe dos dados testados foi estabelecida a maior probabilidade de fazer parte de determinada classe para cada instante, e esses dados foram plotados para melhor visualização.

A Figura 5.15 apresenta a classificação dos dados da perturbação em C_{Af}, aplicada no instante de 2 h e removida no instante de 10 h.

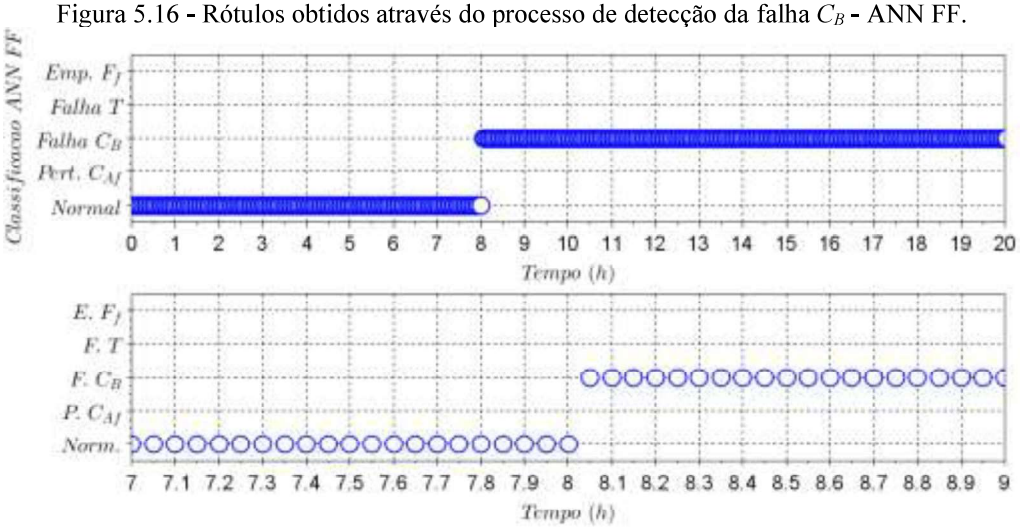
Figura 5.15 - Rótulos obtidos através do processo de detecção da perturbação C_{Af} - ANN FF.



A perturbação passou a ser detectada após um intervalo de amostragem (0,05 h). Após a remoção da perturbação, o sistema de detecção levou um intervalo de amostragem (0,05 h) para detectar novamente a operação normal.

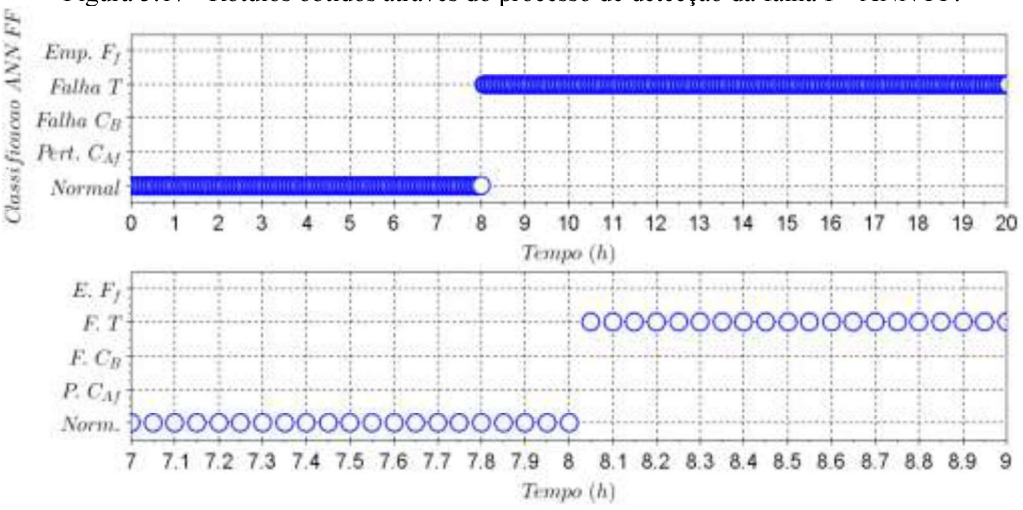
A Figura 5.16 apresenta a classificação dos dados da falha em C_B, aplicada no instante de 8 h. A falha passou a ser detectada instantaneamente.

Figura 5.16 - Rótulos obtidos através do processo de detecção da falha C_B - ANN FF.



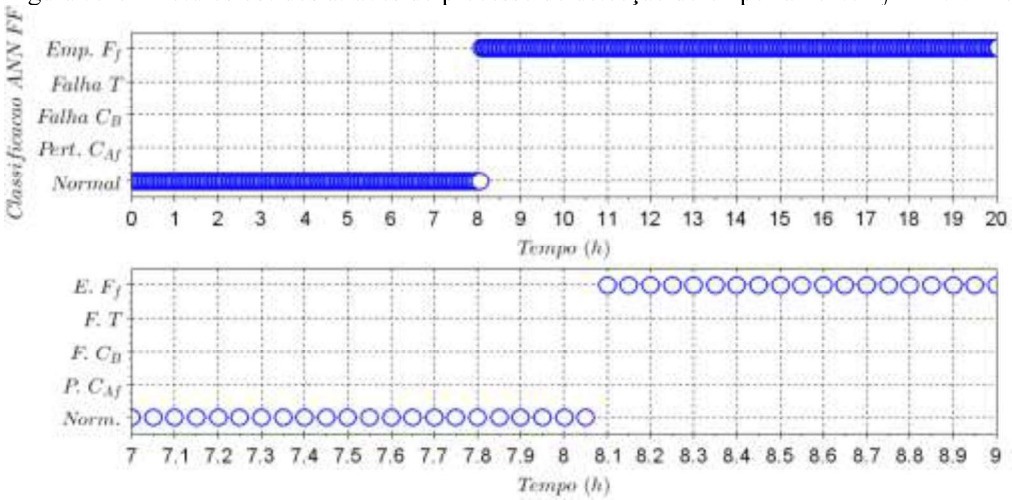
A Figura 5.17 apresenta a classificação dos dados da falha no sensor T, aplicada no instante de 8 h. A falha foi detectada após um intervalo de amostragem (0,05 h).

Figura 5.17 - Rótulos obtidos através do processo de detecção da falha T - ANN FF.



A Figura 5.18 apresenta a classificação dos dados do emperramento em F_f, aplicada no instante de 8 h. A falha foi detectada instantaneamente.

Figura 5.18 - Rótulos obtidos através do processo de detecção do emperramento F_f - ANN FF.



Foram contabilizados todos os instantes nos quais o comportamento do sistema era normal e todos os instantes em que o sistema estava passando pelas diferentes falhas treinadas, de forma que foram criados índices que mostram qual a porcentagem de identificações corretas ou não, de acordo com as Equações (3.1) e (3.2), para todas as operações com falhas testadas para o sistema de detecção e diagnóstico através de ANN FF.

A Tabela 5.1 apresenta os índices encontrados para o sistema de detecção e diagnóstico através do ANN FF com 10 nós na camada oculta.

A Tabela 5.1 apresenta os índices obtidos para a modelagem através de RNA *feedforward*. Para todas as operações estudadas, a ANN FF foi capaz de detectar perfeitamente todos os instantes em que ocorriam falha e todos os instantes em que o processo estava em operação normal. O método de detecção e diagnóstico de falhas através de redes neuronais artificiais com alimentação direta (ANN FF), comparado aos métodos de detecção e diagnóstico de falhas através de máquinas de vetores de suporte (SVC *one-against-one* e SVC *one-against-all*), resulta em detecção e diagnóstico praticamente instantâneos, levando em todos os cenários testados somente um instante para a detecção. Porém, o treinamento de uma rede neuronal artificial, além das dificuldades inerentes a obtenção de sua estrutura otimizada, demanda uma alta capacidade computacional, levando cerca de 4 horas para o seu treinamento.

Tabela 5.1 - Índices encontrados - ANN FF - 4:10:5.

Oper.	Índices - ANN FF - 4:10:5			
	DF/F	DF/N	DN/N	DN/F
Pert. C_{Af}	0,99375	0,00417	0,99583	0,00625
Falha C_B	1,00000	0,00	1,00000	0,00
Falha T	1,00000	0,00	1,00000	0,00
Emp. F_f	0,99583	0,00	1,00000	0,00417

Presença de ruídos com maior intensidade - ANN FF

Para confirmar a eficácia do modelo de detecção e diagnóstico de falhas através do ANN de alimentação direta, decidiu-se realizar a detecção novamente em dados que apresentavam maior ruído do que os dados utilizados no treinamento do sistema. Os sinais com 10 vezes maiores ruídos utilizados na seção sobre SVM foram utilizados novamente para efeitos de comparação.

O sistema de detecção e diagnóstico através do ANN FF, na grande maioria de operações realizadas com maior ruído, apresentou um intervalo de amostragem em que a ANN FF detectou um tipo de falha e também a operação normal. Apesar desta detecção errônea, a rede neuronal é capaz de detectar muito bem os dados amostrais perante os dados treinados. A modificação de ruídos não foi capaz de degradar o sistema de detecção de falhas de modo acentuado, como é possível verificar os rótulos obtidos para as amostras com ruídos 10 vezes maiores para a Perturbação em C_{Af}, Falha em C_B, Falha em T e Emperramento de F, respectivamente nas Figuras Figura 5.19 a Figura 5.22.

Figura 5.19 - Rótulos obtidos através do processo de detecção da perturbação C_{Af} - ANN FF - ruído 10x.

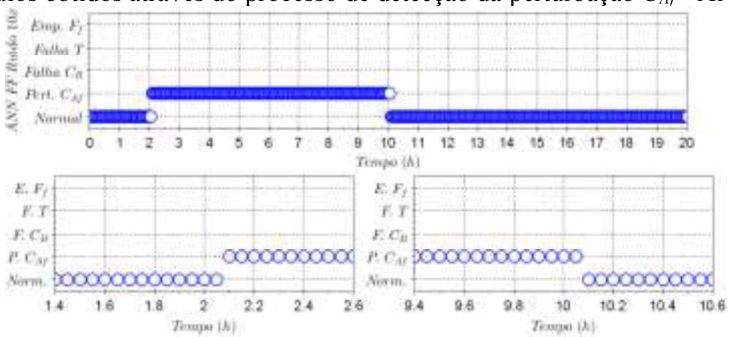


Figura 5.20 - Rótulos obtidos através do processo de detecção da falha C_B - ANN FF - ruído 10x.

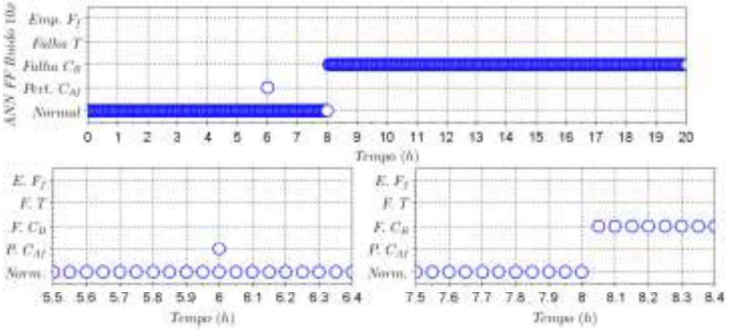


Figura 5.21 - Rótulos obtidos através do processo de detecção da falha *T* - ANN FF - ruído 10x.

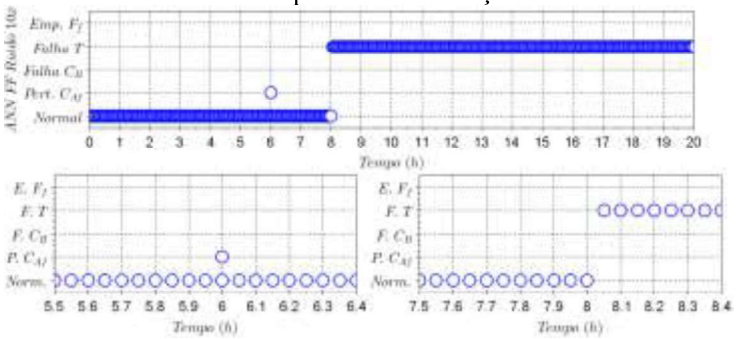
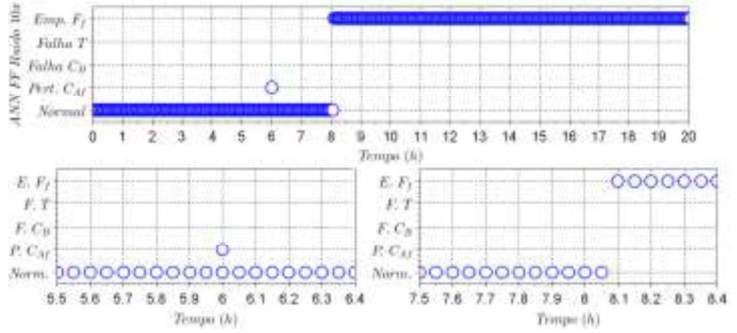


Figura 5.22 - Rótulos obtidos através do processo de detecção do emperramento F_f - ANN FF - Ruído 10x.



A Tabela 5.2 apresenta os índices encontrados para o sistema de detecção e diagnóstico através do ANN FF com ruídos 10x maiores.

Tabela 5.2 - Índices encontrados - ANN FF - Ruídos 10x maiores - 4:10:5.

Oper.	Índices - ANN FF - Ruídos 10x maiores - 4:10:5			
	DF/F	DF/N	DN/N	DN/F
Pert. C_{Af}	0,99375	0,00417	0,99583	0,00625
Falha C_B	1,00000	0,00000	1,00000	0,00000
Falha T	1,00000	0,00000	1,00000	0,00000
Emp. F	0,99583	0,00000	1,00000	0,00417

Pode se perceber pelas Figuras Figura 5.19 a Figura 5.22 e pela Tabela 5.2 que mesmo com dados amostrais com ruídos 10 vezes maiores, o modelo ANN FF treinado com os dados originais conseguiu manter a precisão na detecção e diagnóstico de falhas, se comparado com a metodologia FDD apresentado anteriormente.

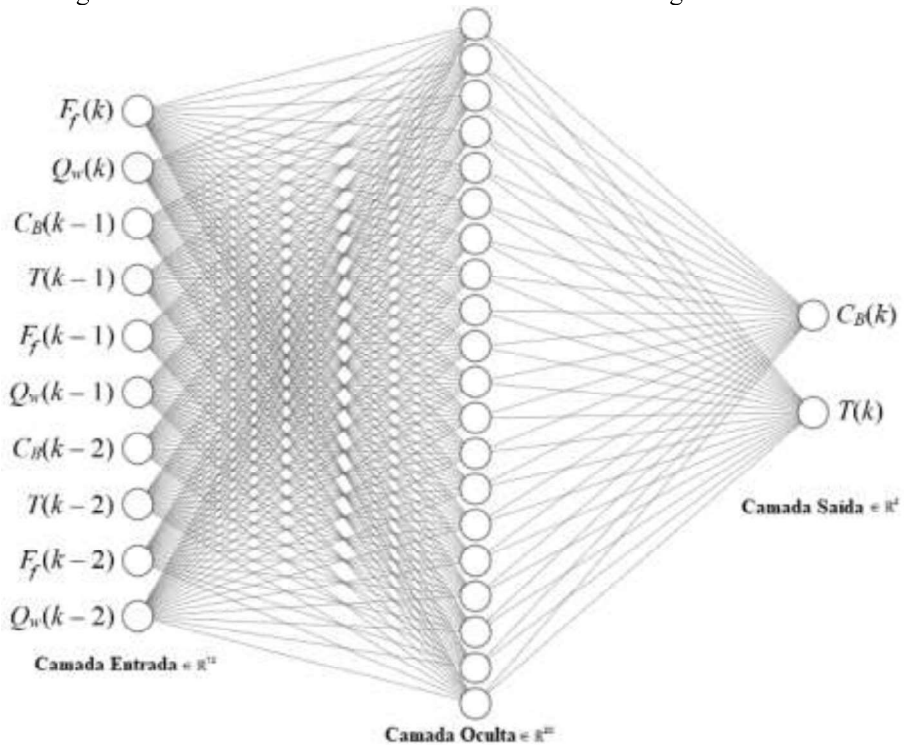
5.10.2. Rede Neuronal Artificial Supervisionada de Regressão - ANN-R - Uma camada oculta

Para o treinamento da rede neuronal artificial de regressão com uma camada oculta, da mesma forma que para a máquina de vetores de suporte de regressão, foi utilizado 2 terços dos dados da operação normal. Com o restante, um terço, foi realizada a validação do modelo ANN-R. Os dados foram selecionados de acordo com a Seção 3.4. A estrutura da ANN-R utilizada

para a detecção de falhas foi estabelecida com uma única camada oculta, cuja quantidade de nós é o dobro da quantidade de nós da camada de entrada da ANN-R.

A entrada da rede neuronal foi definido como um nó para cada uma das variáveis controladas e manipuladas (C_B, T, F_f e $\dot{Q}_w$), em instantes anteriores (k, $k-1$, $k-2$), totalizando 10 nós. Foi estabelecida uma camada interna, também chamada de camada oculta, totalizando uma rede 10:20:2. A camada de saída da rede neuronal será a resposta do modelo, as variáveis controladas no instante k (C_B e T), de acordo com a Figura 5.23.

Figura 5.23 - Estrutura da Rede Neuronal Artificial de Regressão: 10:20:2.



Para o treinamento da rede, foram estabelecidas três mil (3000) épocas, após estudo que identificou que o erro médio quadrático (*Median Square* Error, MSE) entre os dados de validação e os dados reais não diminuía com a mesma intensidade. Cada época no processo de treinamento da rede neuronal é utilizada para melhorar a matriz de pesos **W**, de acordo com a Equação (5.24). A taxa de aprendizado utilizado foi de 0,9, escolhida de maneira empírica.

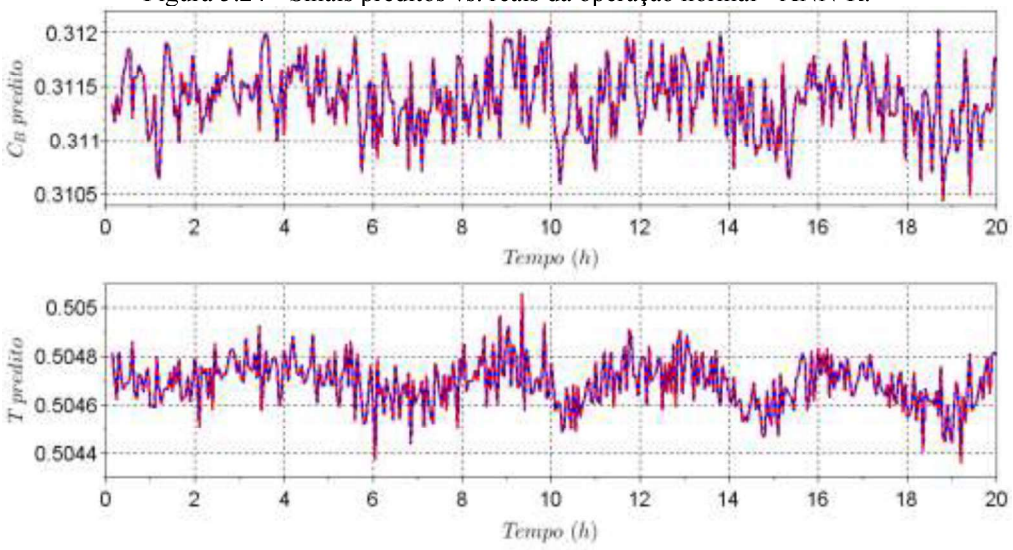
Após o treinamento e validação da rede, a matriz peso **W** foi armazenada, assim como as informações pertinentes à rede ANN de regressão, como a estrutura da rede e a taxa de aprendizado cuja rede foi treinada. Posteriormente, estas informações foram utilizadas para a aplicação da rede neuronal treinada nos dados faltosos. A saída da rede é o valor para as variáveis controladas (C_B e T) predito pelo modelo, que são comparados com o valor real do sistema. Caso o desvio entre o valor real e o valor predito das variáveis for maior que três vezes a variância dos dados das variáveis controladas da operação normal, é dito que o sistema está

passando por uma operação faltosa (critério de verificação de limites).

Foi monitorada a Estatística T^2 de Hotelling para todas as operações testadas pelo sistema de detecção de falhas com ANN de regressão.

A Figura 5.24 apresenta a resposta da rede neural de regressão treinada e validada com os dados da operação normal normalizados.

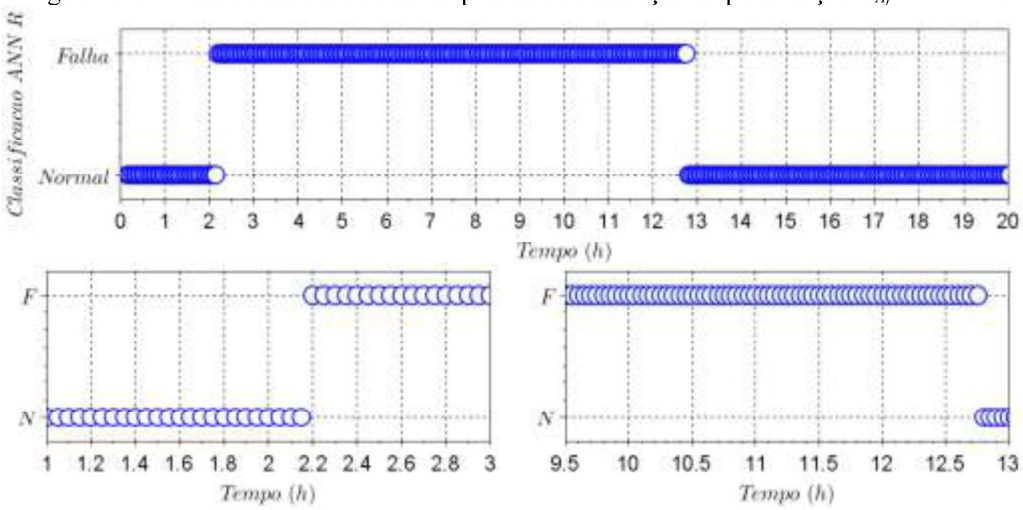
Figura 5.24 - Sinais preditos vs. reais da operação normal - ANN-R.



A Figura 5.25 apresenta os rótulos obtidos através do modelo ANN-R para detecção de falhas do sinal do sistema passando pela Perturbação C_{Af}, do instante 2 h ao instante 10 h. No ANN-R, devido ao modelo ser de regressão, o sistema de detecção de falhas é capaz só de detectar qualquer operação diferente da operação normal.

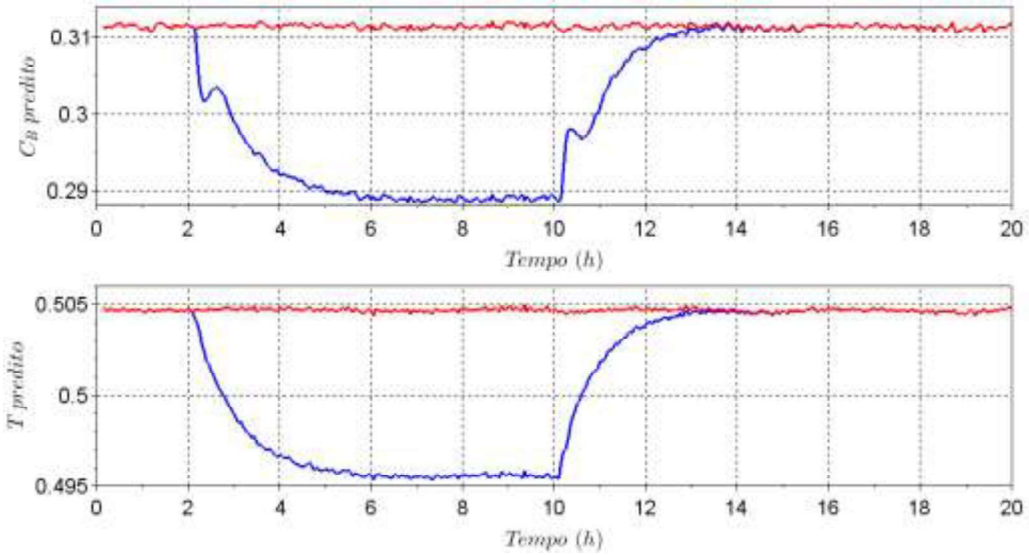
O comportamento do sistema de detecção de falhas através do ANN-R mostra que a detecção da perturbação aconteceu após 3 intervalos de amostragem (0,15 h), porém o sistema não foi rápido para voltar a operação normal, após a remoção da perturbação em C_{Af}. O sistema levou 55 intervalos de amostragem (2,75 h) para retornar à detecção da operação normal. A dinâmica apresentada pelas variáveis controladas (C_B e T) explica o tempo que o sistema de detecção de falhas levou para informar a modificação da concentração de A.

Figura 5.25 - Rótulos obtidos através do processo de detecção da perturbação C_{Af} - ANN-R.



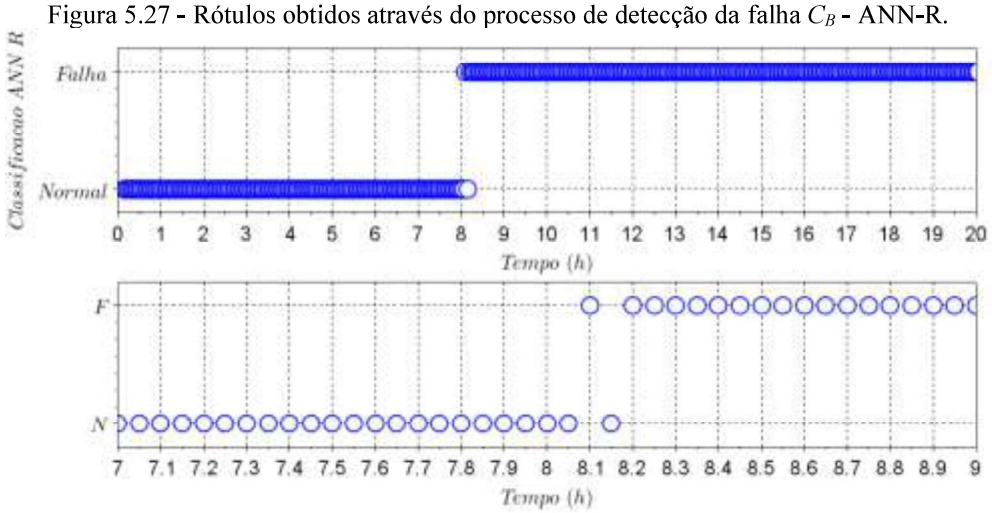
A Figura 5.26 apresenta o sinal predito em comparação com o sinal da operação normal, para as variáveis controladas.

Figura 5.26 - Sinais preditos operação normal vs. falha - perturbação C_{Af} - ANN-R.



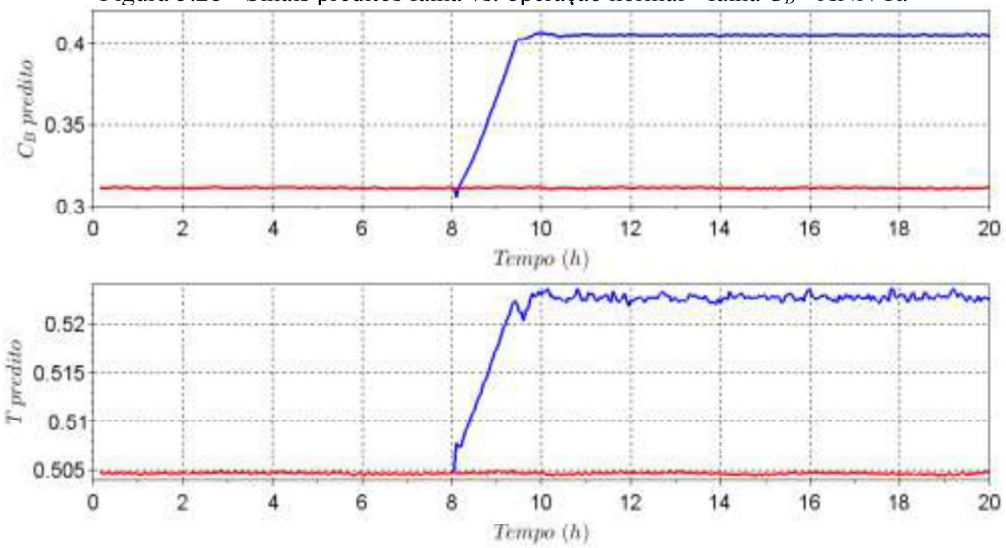
A Figura 5.27 apresenta os rótulos obtidos através do sistema de detecção utilizando o modelo ANN-R para o sinal do sistema reacional passando pela falha C_B, a partir do instante 8 h. O sistema de detecção levou 1 intervalo de amostragem (0,05 h) para detectar pela primeira vez que o sistema passou de uma operação normal para uma operação faltosa, porém, no instante seguinte (8,15 h), após detectar por um instante a falha, o sistema de detecção voltou a detectar a operação normal. Finalmente, um intervalo de amostragem após a detecção errada, o sistema voltou a detectar a falha.

Figura 5.27 - Rótulos obtidos através do processo de detecção da falha C_B - ANN-R.



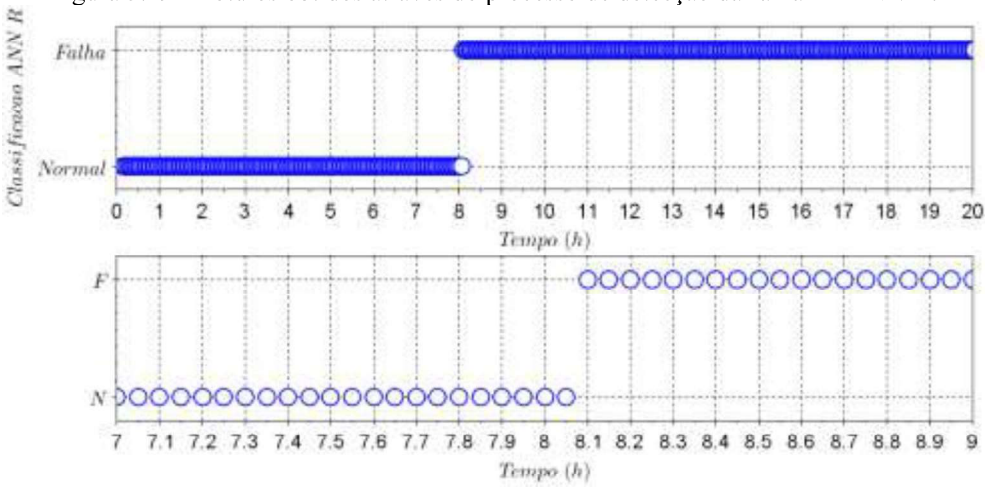
A Figura 5.28 apresenta o sinal predito em comparação com o sinal da operação normal, para as variáveis controladas durante a Falha C_B.

Figura 5.28 - Sinais preditos falha vs. operação normal - falha C_B - ANN-R.



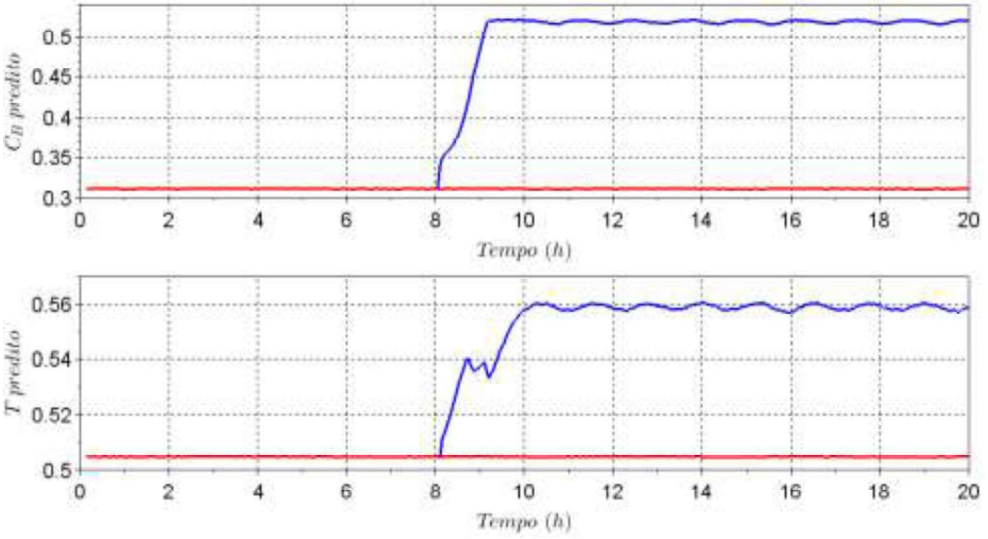
A Figura 5.29 apresenta os rótulos obtidos através do sistema de detecção utilizando o método de detecção de falhas através de ANN-R para o sinal do sistema reacional passando pela falha *T*, a partir do instante 8 h. O sistema de detecção levou 1 intervalo de amostragem (0,05 h) para detectar que o sistema passou de uma operação normal para uma operação faltosa.

Figura 5.29 - Rótulos obtidos através do processo de detecção da falha *T* - ANN-R.



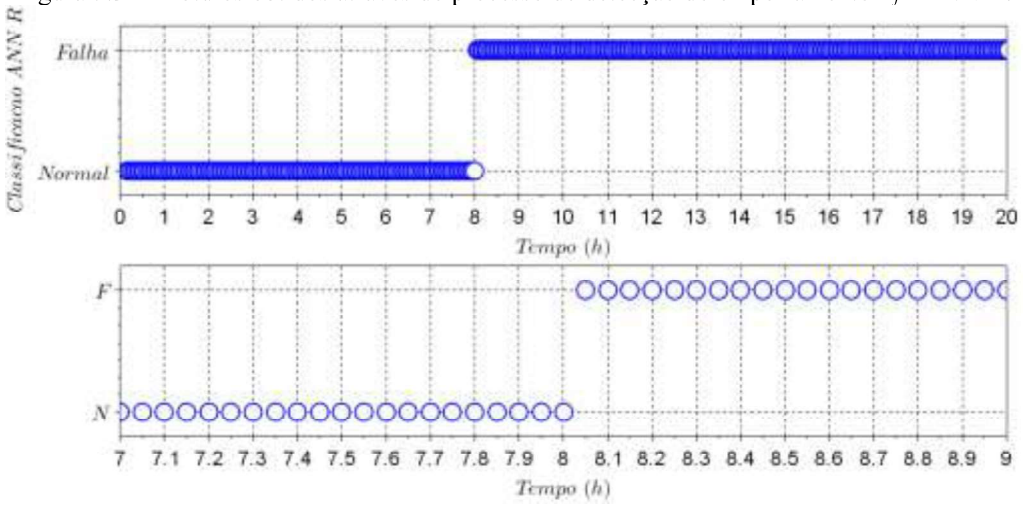
A Figura 5.30 apresenta o sinal predito em comparação com o sinal da operação normal, para as variáveis controladas.

Figura 5.30 - Sinais preditos falha vs. operação normal - falha *T* - ANN-R.



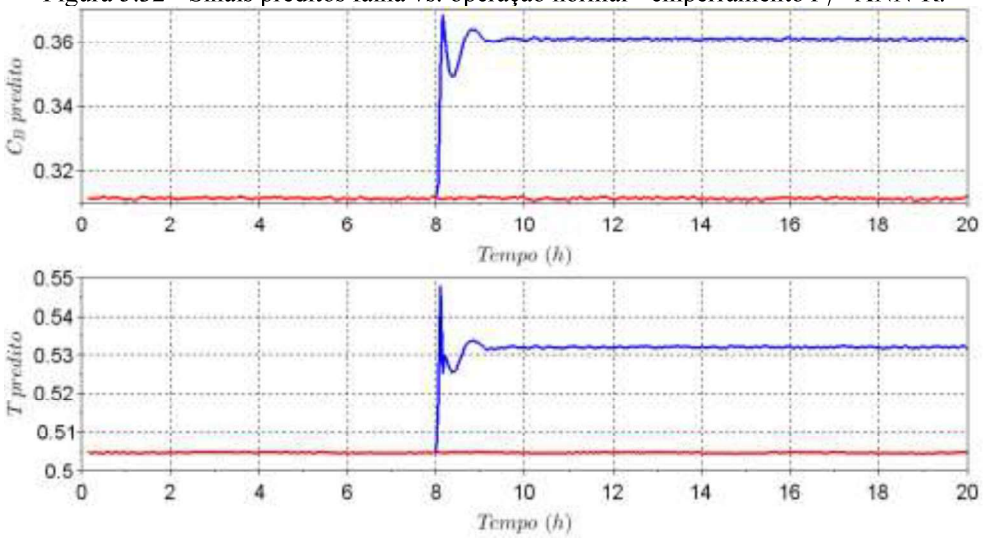
A Figura 5.31 apresenta os rótulos obtidos através do sistema de detecção de falhas utilizando a rede neuronal de regressão para o sinal do sistema reacional passando pelo Emperramento de *F*, a partir do instante 8 h.

Figura 5.31 - Rótulos obtidos através do processo de detecção do emperramento F_f - ANN-R.



O sistema de detecção agiu instantaneamente, no intervalo de amostragem seguinte a aplicação da falha, para detectar que o sistema passou de uma operação normal para uma operação faltosa. A Figura 5.32 apresenta o sinal predito em comparação com o sinal da operação normal, para as variáveis controladas.

Figura 5.32 - Sinais preditos falha vs. operação normal - emperramento F_f - ANN-R.



A Tabela 5.3 apresenta os índices da Equação (3.1) calculados para o sistema de detecção de falhas através da rede neuronal artificial de regressão, o ANN-R.

Tabela 5.3 - Índices encontrados - ANN-R - Única camada oculta.

Oper.	Índices - ANN-R - 10:20:2			
	DF/F	DF/N	DN/N	DN/F
Pert. C_{Af}	0,98125	0,2292	0,7708	0,01875
Falha C_B	0,99167	0,0000	1,0000	0,00833
Falha T	0,99583	0,0000	1,0000	0,00417
Emp. F	1,0000	0,0000	1,0000	0,0000

O método de detecção de falhas através da ANN-R não possui a capacidade de diagnosticar a operação faltosa e possui certas dificuldades a serem vencidas, como a seleção da melhor estrutura da rede e a melhor configuração de treinamento. Além disso, o ANN-R necessita maior capacidade computacional para o treinamento que os métodos estatísticos de máquinas de vetores de suporte, apesar do tempo de treinamento ser menor que o outro método utilizando redes neuronais, a ANN FF. No presente estudo de caso, a rede neuronal de regressão com uma única camada foi capaz de detectar todas as operações faltosas em menos de três intervalos de amostragem (0,15 h) e não conseguir retornar à operação normal de uma forma eficiente, no caso da perturbação C_{Af} (durante 2,75 h o sistema de detecção não foi capaz de detectar novamente a operação normal).

A rede neuronal artificial de regressão mostrou-se um método preciso para detectar a operação anormal, apesar de em uma das situações avaliadas (perturbação C_{Af}) o sistema ter demorado a detectar novamente a operação normal. Isto ocorre devido as dinâmicas das variáveis de sistema selecionadas para análise.

5.10.3. Rede Neuronal Artificial Supervisionada de Regressão - ANN-R - Duas Camadas Ocultas

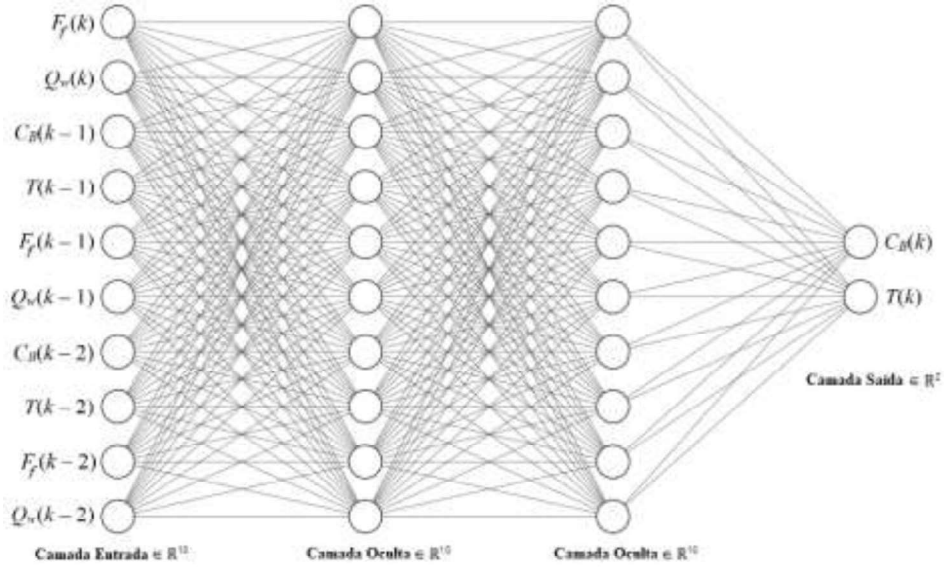
A estrutura da ANN-R utilizada para a detecção de falhas nesta seção foi estabelecida com duas camadas ocultas, cada uma delas com o mesmo número de nós da camada de entrada. A entrada da rede neuronal foi definida como um nó para cada uma das variáveis controladas e manipuladas (C_B, T, F_f e $\dot{Q}_w$), em instantes anteriores (k, $k-1$, $k-2$), totalizando 10 nós. Foram estabelecidas 2 camadas internas, também chamadas de camadas ocultas, totalizando uma rede 10:10:10:2. A camada de saída da rede neuronal é a resposta do modelo, as variáveis controladas no instante k (C_B e T), de acordo com a Figura 5.33.

Para o treinamento da rede, foram estabelecidas três mil (3000) épocas, após estudo que identificou que o erro médio quadrático (*Median Square* Error, MSE) entre os dados de validação e os dados reais não diminuía após este número de épocas. Cada época no processo de treinamento da rede neuronal é utilizada para melhorar a matriz de pesos **W**, de acordo com a Equação (5.24). A taxa de aprendizado empírica utilizada foi de 0,9, a mesma utilizada para todos os métodos de detecção de falhas através de redes neuronais artificiais.

Após o treinamento e validação da rede, a matriz peso **W** e as informações pertinentes à rede ANN de regressão foram armazenadas, tais quais a estrutura das camadas da rede e a taxa de aprendizado cuja rede foi treinada. De posse destas informações, posteriormente foi aplicada a rede neuronal treinada aos dados faltosos. A saída da rede é o valor para as variáveis controladas (C_B e T) predito pelo modelo, que são comparados com o valor real do sistema.

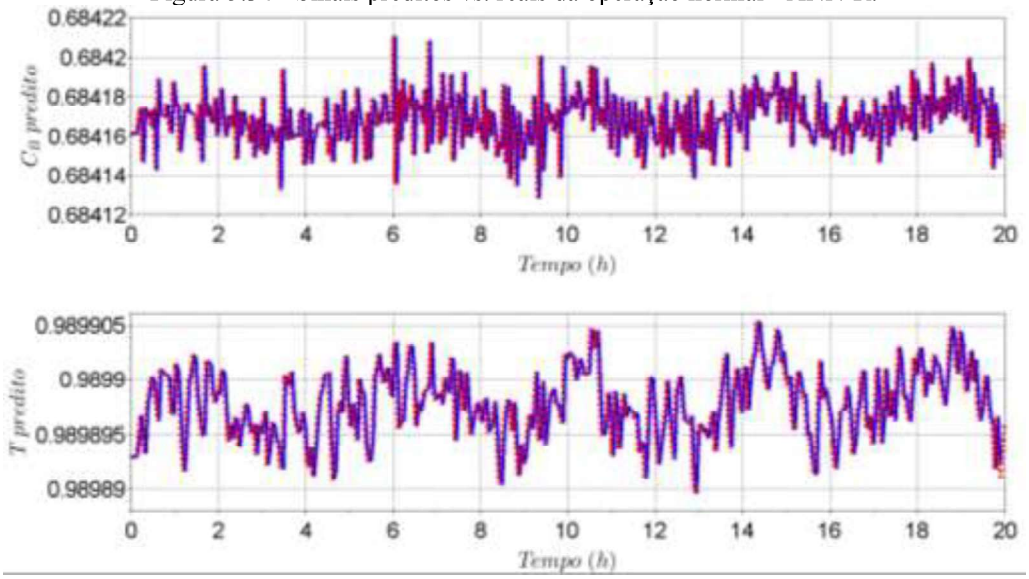
Caso o desvio entre o valor real e o valor predito das variáveis for maior que três vezes a variância dos dados das variáveis controladas da operação normal, é dito que o sistema está passando por uma operação faltosa (critério da variância).

Figura 5.33 - Estrutura da Rede Neuronal Artificial de Regressão: 10:10:10:2.



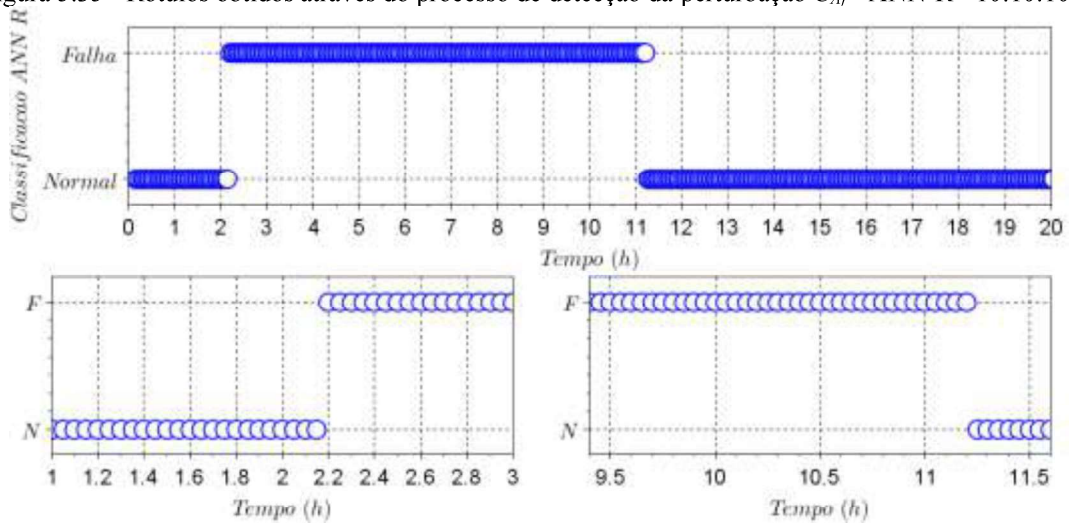
A Figura 5.34 apresenta a resposta da rede neural de regressão treinada e validada com os dados da operação normal normalizados.

Figura 5.34 - Sinais preditos vs. reais da operação normal - ANN-R.



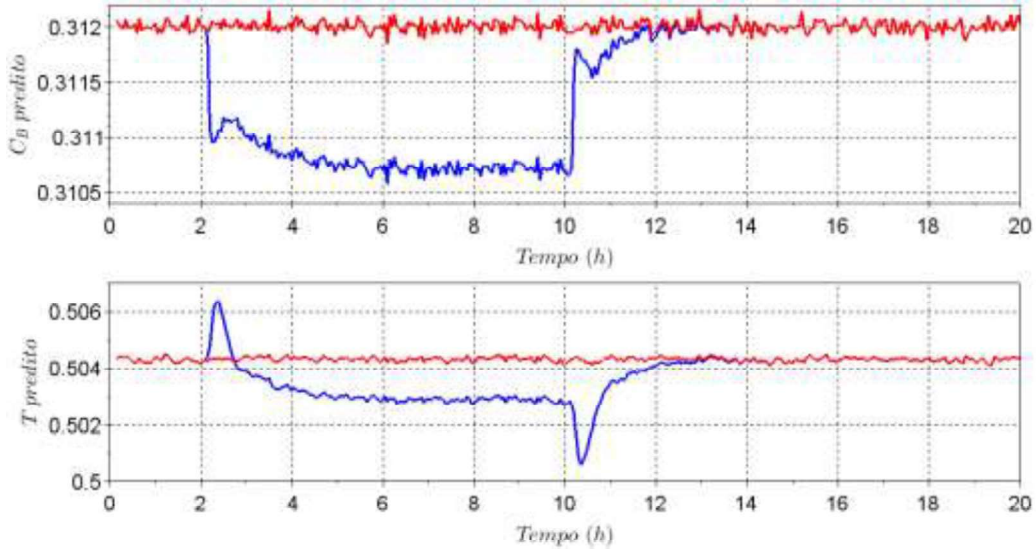
A Figura 5.25 apresenta os rótulos obtidos através do modelo ANN-R para detecção de falhas do sinal do sistema passando pela perturbação C_{Af}, do instante 2 h ao instante 10 h. No ANN-R, devido ao modelo ser de regressão, o sistema de detecção de falhas é capaz só de detectar qualquer operação diferente da operação normal.

Figura 5.35 - Rótulos obtidos através do processo de detecção da perturbação C_{Af} - ANN-R - 10:10:10:2.



O comportamento do sistema de detecção de falhas através do ANN-R mostra que a detecção da perturbação ocorreu após 1 intervalo de amostragem (0,05 h), porém o sistema não foi rápido para voltar a operação normal, após a remoção da perturbação em C_{Af}. O sistema levou 24 intervalos de amostragem (1,2 h) para retornar à detecção da operação normal. A dinâmica apresentada pelas variáveis controladas (C_B e T) explica o tempo que o sistema de detecção de falhas levou para informar a modificação da concentração de A. A Figura 5.36 apresenta os sinais preditos comparados com os sinais da operação normal para as variáveis controladas C_B e T.

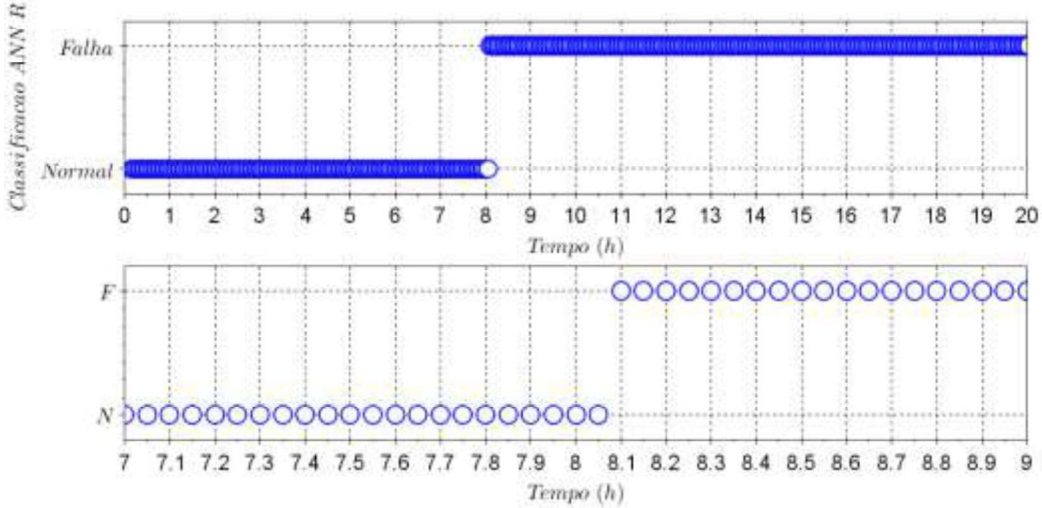
Figura 5.36 - Sinais preditos operação normal vs. falha - perturbação C_{Af} - ANN-R - 10:10:10:2.



A Figura 5.37 apresenta os rótulos obtidos através do sistema de detecção utilizando o modelo ANN-R para o sinal do sistema reacional passando pela Falha C_B, a partir do instante 8 h. O sistema de detecção levou 1 intervalo de amostragem (0,05 h) para detectar que o sistema

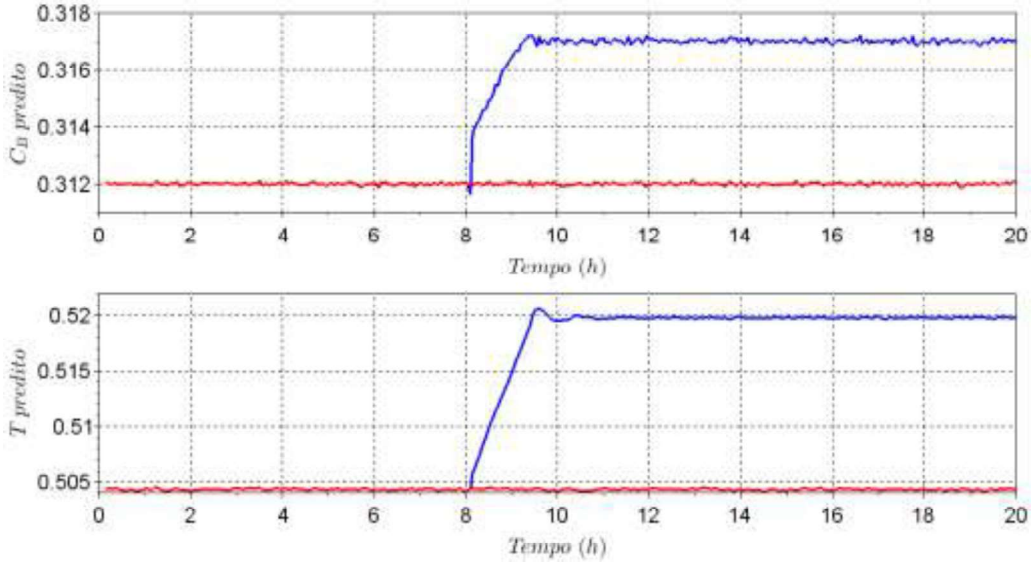
passou de uma operação normal para uma operação faltosa.

Figura 5.37 - Rótulos obtidos através do processo de detecção da falha C_B - ANN-R - 10:10:10:2.



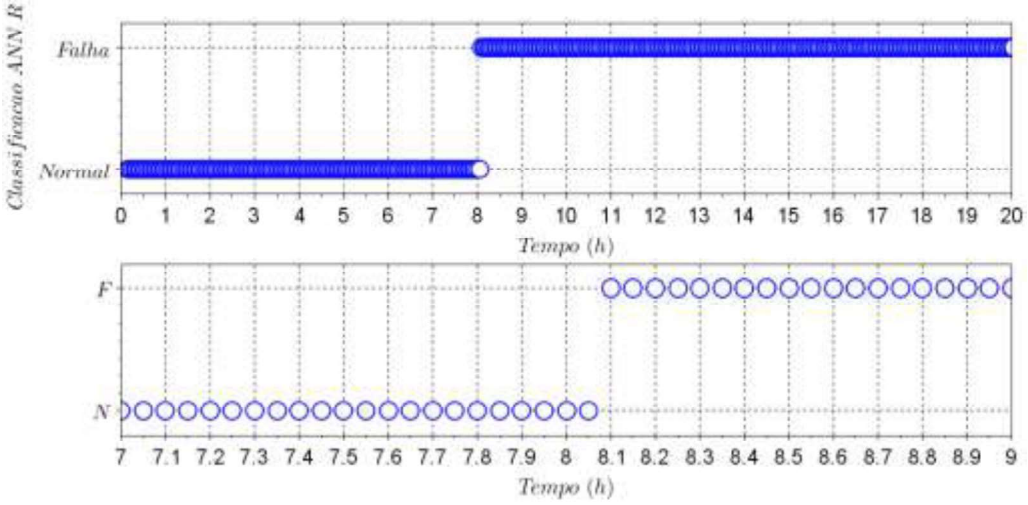
A Figura 5.38 apresenta os sinais preditos em comparação com os sinais da operação normal das variáveis controladas durante a falha C_B.

Figura 5.38 - Sinais preditos falha vs. operação normal - falha C_B - ANN-R - 10:10:10:2.



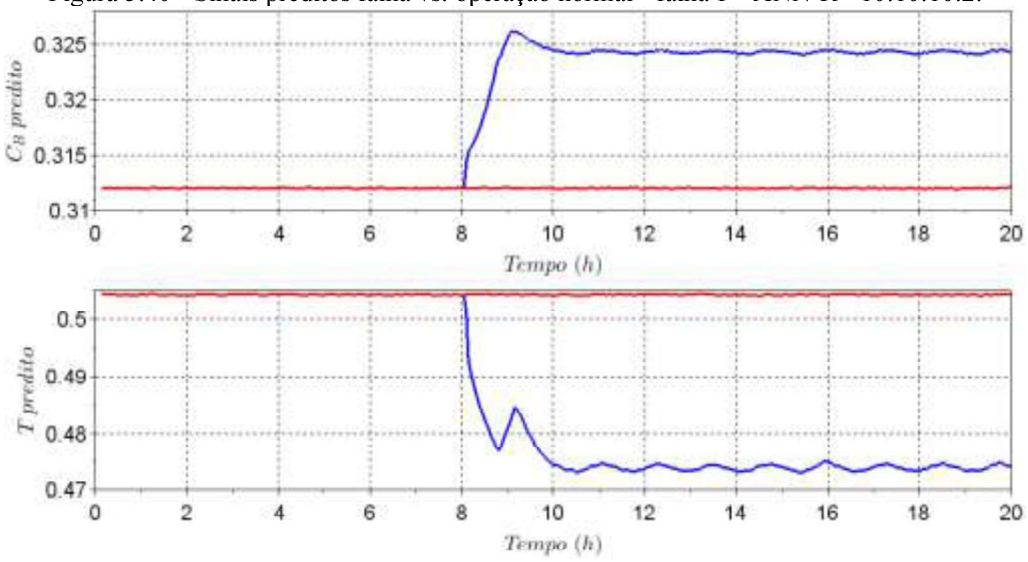
A Figura 5.39 apresenta os rótulos obtidos através do sistema de detecção utilizando o método de detecção de falhas através de ANN-R para o sinal do sistema reacional passando pela falha *T*, a partir do instante 8 h. A ANN-R levou 1 intervalo de amostragem (0,05 h) para detectar que o sistema passou de uma operação normal para uma operação faltosa.

Figura 5.39 - Rótulos obtidos através do processo de detecção da falha *T* - ANN-R - 10:10:10:2.



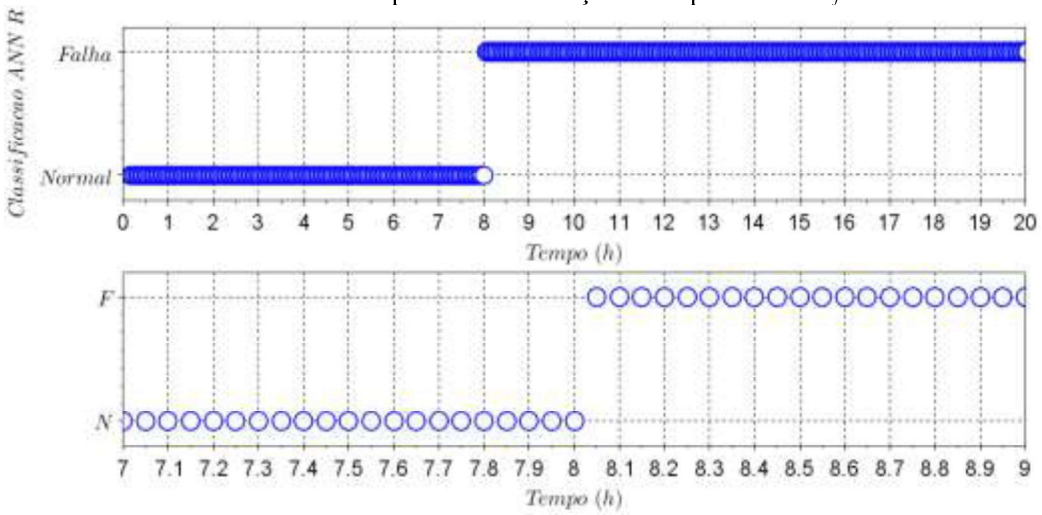
A Figura 5.40 apresenta o sinal predito em comparação com o sinal da operação normal para a falha *T* das variáveis controladas C_B e *T*.

Figura 5.40 - Sinais preditos falha vs. operação normal - falha *T* - ANN-R - 10:10:10:2.



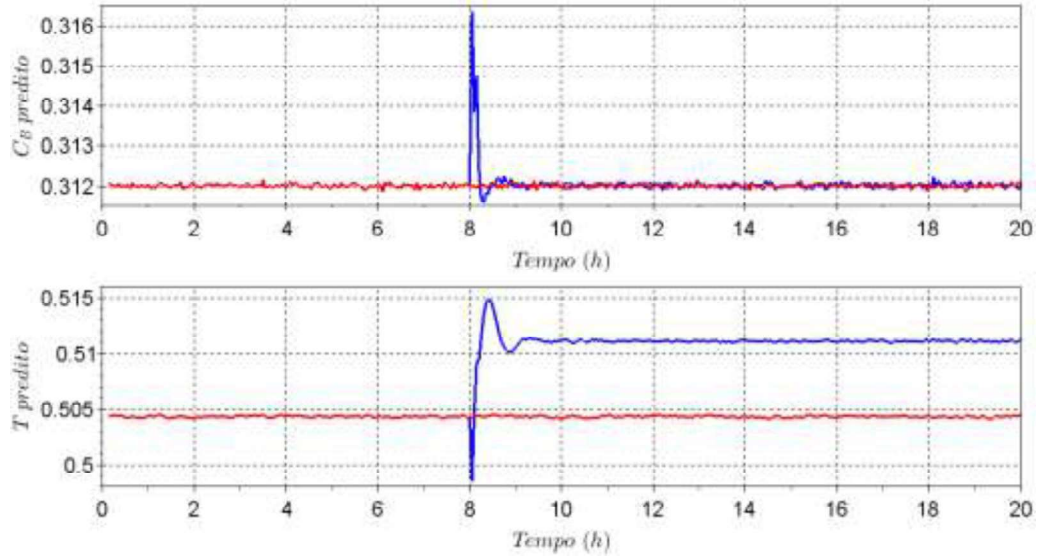
A Figura 5.41 apresenta os rótulos obtidos através do sistema de detecção de falhas utilizando a rede neuronal de regressão para o sinal do sistema reacional passando pelo Emperramento de *F*, a partir do instante 8 h.

Figura 5.41 - Rótulos obtidos através do processo de detecção do emperramento F_f - ANN-R - 10:10:10:2.



O modelo de redes neuronais artificiais de regressão, com duas camadas ocultas, detecta instantaneamente que o sistema passou de uma operação normal para uma operação faltosa. A Figura 5.42 apresenta o sinal predito em comparação com o sinal da operação normal, para as variáveis controladas.

Figura 5.42 - Sinais preditos falha vs. operação normal - emperramento F_f - ANN-R - 10:10:10:2.



A Tabela 5.4 apresenta os índices da Equação (3.1) calculados para o sistema de detecção de falhas através da rede neuronal artificial de regressão, o ANN-R, com estrutura 10:10:10:2.

Tabela 5.4 - Índices encontrados - ANN-R - 10:10:10:2.

Oper.	Índices - ANN-R - 10:10:10:2			
	DF/F	DF/N	DN/N	DN/F
Pert. C_{Af}	0,98125	0,1000	0,9000	0,01875
Falha C_B	0,99583	0,0000	1,0000	0,00417
Falha T	0,99583	0,0000	1,0000	0,00417
Emp. F	1,0000	0,0000	1,0000	0,0000

Apesar do método de detecção de falhas através da ANN-R não diagnosticar a operação faltosa, a detecção da falha ocorre de maneira satisfatória, gastando em torno de 1 intervalo de amostragem para a detecção da falha. Porém, o método apresenta certas dificuldades a serem vencidas, como a seleção da melhor estrutura da rede e a melhor configuração dos parâmetros de treinamento (taxa de aprendizagem e número de épocas, por exemplo). Além disso, a ANN-R necessita uma maior capacidade computacional para o treinamento que os métodos estatísticos de máquinas de vetores de suporte. A ANN-R, por não necessitar das informações de falhas para o treinamento, demanda menor capacidade computacional o método de redes neuronais de classificação, como a ANN FF. No presente estudo de caso, a rede neuronal de regressão foi capaz de detectar todas as operações faltosas dentro de um intervalo de amostragem (0,05 h) e não conseguiu retornar à operação normal, no caso da Perturbação C_{Af}, de uma forma eficiente (durante 1,2 h o sistema de detecção não foi capaz de detectar novamente a operação normal.

A rede neuronal artificial de regressão mostrou-se um método eficiente para a detecção de operações anormais, porém em uma das situações avaliadas (perturbação em C_{Af}) o sistema demorou a detectar novamente a operação normal, podendo prejudicar as decisões tomadas pelos operadores. Isto ocorre as dinâmicas das variáveis de sistema selecionadas para o sistema de detecção de falhas.

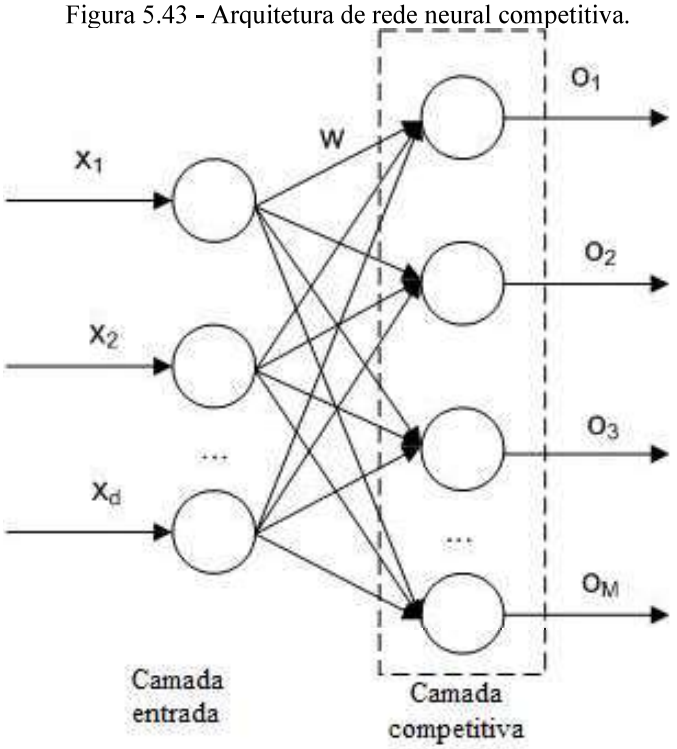
5.10.4. Rede Neuronal Artificial Não-Supervisionada *Winner-Take-All*

As redes neuronais artificiais de aprendizagem não-supervisionada são redes em que não existe a necessidade de fornecer a saída das mesmas durante a fase de treinamento. A rede é construída através das amostras das variáveis de entrada e a quantidade de nós (classes) da própria rede. No caso das RNAs *Winner-Take-All*, elas são muito utilizadas para reconhecimento de padrões.

A entrada da rede neuronal foi definida como um nó para cada uma das variáveis controladas e manipuladas (C_B, T, F_f e $\dot{Q}_w$) no instante k, totalizando 4 nós. Foram estabelecidas 5 classes (ou nós) de saída para a rede WTA. A Rede Neuronal Artificial WTA tem a capacidade de extrair características importantes dos dados de treinamento que possam classificar a massa

de dados. A camada de saída da rede neuronal apresentará um vetor de com os neurônios ativados (1) e desativados (0) de cada dado amostrado. A Figura 5.43 apresenta a arquitetura de uma rede *winner-take-all*. No caso ilustrado, a estrutura de saída da rede WTA foi de cinco nós, de acordo com o número de classes de detecção.

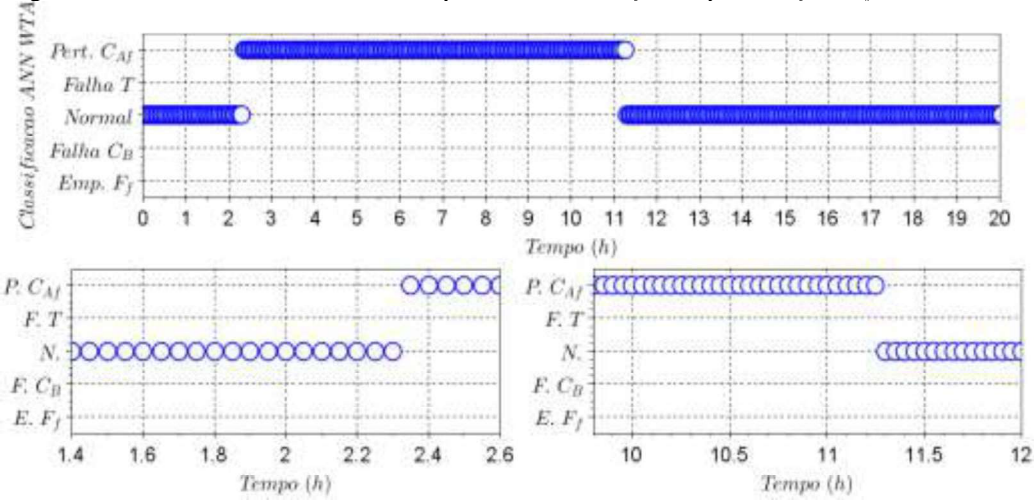
Figura 5.43 - Arquitetura de rede neural competitiva.



Fonte: SALATAS, 2011.

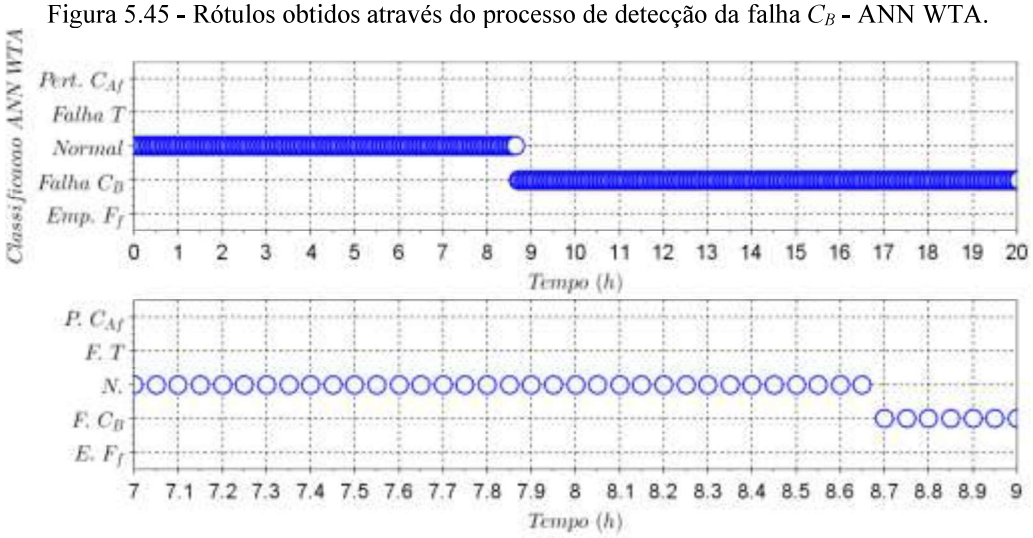
A Figura 5.44 apresenta a classificação dos dados da perturbação em C_{Af}, aplicada no instante de 2 h e removida no instante de 10 h. A perturbação passou a ser detectada após 6 intervalos de amostragem (0,30 h). Após a remoção da perturbação, o sistema de detecção levou 25 intervalos de amostragem (1,25 h) para detectar novamente a operação normal.

Figura 5.44 - Rótulos obtidos através do processo de detecção da perturbação C_{Af} - ANN WTA.



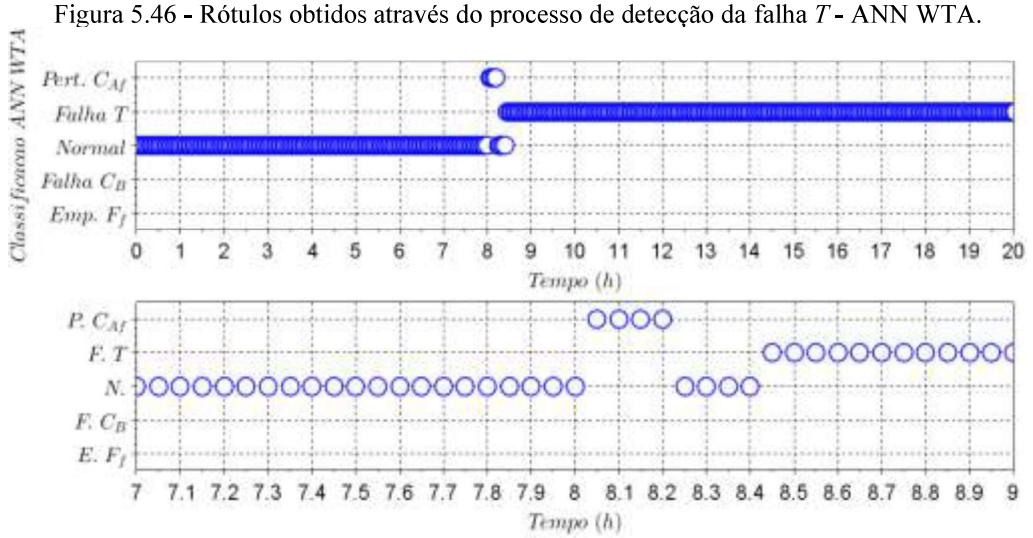
A Figura 5.45 apresenta a classificação dos dados da falha em C_B, aplicada no instante de 8 h. A falha passou a ser detectada após 13 intervalos de amostragem (0,65 h).

Figura 5.45 - Rótulos obtidos através do processo de detecção da falha C_B - ANN WTA.



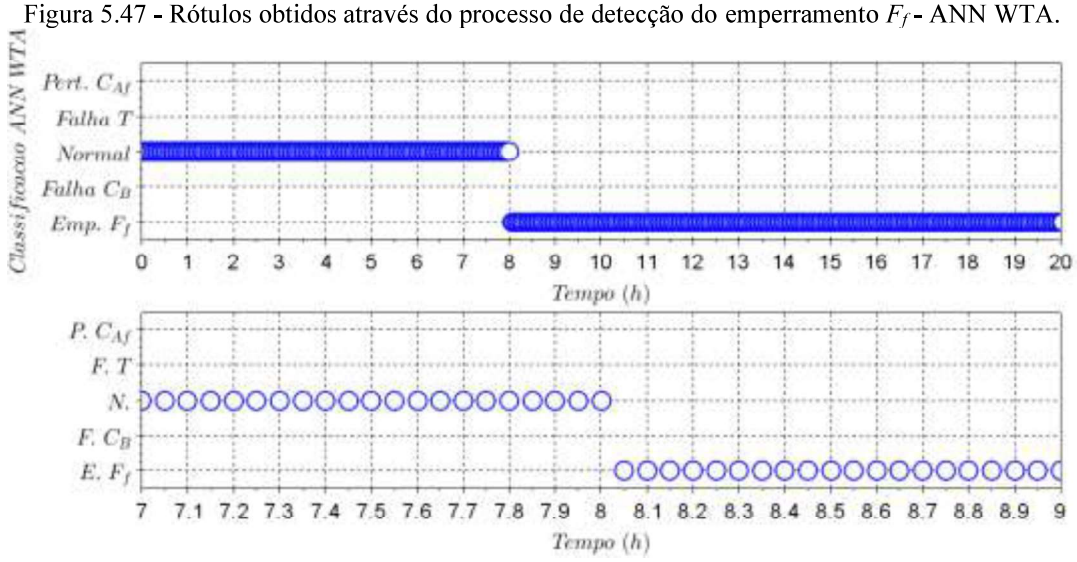
A Figura 5.46 apresenta a classificação dos dados da falha no sensor T, aplicada no instante de 8 h. O ANN WTA cometeu alguns erros de classificação. No primeiro momento, durante 4 intervalos de amostragem (0,20 h), o sistema detectou a perturbação C_{Af}, retornando a detecção da operação normal durante mais 4 intervalos de amostragem (0,20 h). Após o instante 8,45 h, o sistema detectou corretamente a falha T.

Figura 5.46 - Rótulos obtidos através do processo de detecção da falha T - ANN WTA.



A Figura 5.47 apresenta a classificação dos dados do emperramento em F_f, aplicada no instante de 8 h. A falha foi detectada instantaneamente.

Figura 5.47 - Rótulos obtidos através do processo de detecção do emperramento F_f - ANN WTA.



Foram contabilizados todos os instantes nos quais o comportamento do sistema era normal e todos os instantes em que o sistema estava passando pelas diferentes falhas treinadas, de forma que foram criados índices que mostram qual a porcentagem de identificações corretas ou não, de acordo com as Equações (3.1) e (3.2), para todas as operações com falhas testadas para o sistema de detecção e diagnóstico através de ANN WTA.

A Tabela 5.5 apresenta os índices encontrados para o sistema de detecção e diagnóstico através do ANN WTA.

Tabela 5.5 - Índices encontrados - ANN WTA.

Oper.	Índices - ANN WTA				
	DF/F	DF/N	DN/N	DN/F	DF_i/F_j
Pert. C_{Af}	0,9625	0,1042	0,8958	0,0375	0,00
Falha C_B	0,9458	0,0000	1,0000	0,0542	0,00
Falha T	0,9666	0,0000	1,0000	0,0167	0,0167
Emp. F	1,0000	0,0000	1,0000	0,0000	0,00

O método de detecção e diagnóstico de falhas através de redes neuronais artificiais *winner-take-all* (ANN WTA) com aprendizagem não-supervisionada, comparado aos métodos de detecção e diagnóstico de falhas através de redes neuronais artificiais de classificação (ANN FF), resulta em detecção e diagnóstico consistentes, apesar de em algumas situações levar alguns instantes a mais para detectar a falha. Porém, o treinamento de uma rede neuronal artificial não supervisionada *winner-take-all* (ANN WTA) demandou menor capacidade computacional que o treinamento da ANN de classificação com alimentação direta. A ANN WTA levou cerca de 10 minutos para ser treinada, enquanto que a ANN FF levou 4 horas para ser treinada. Além disso, a ANN FF necessita dos dados de treinamento rotulados, enquanto que a ANN WTA é treinada somente com os dados de entrada da rede neuronal.

5.10.5. Discussão

A análise dos métodos de detecção e diagnóstico de falhas com utilização de RNAs foi realizada para verificar as características positivas e negativas de cada método. A Tabela 5.6 apresenta os valores dos índices apresentados na Seção 3.3, em que foram baseadas as conclusões a respeito dos métodos SVM.

Tabela 5.6 - Índices - ANN FF, ANN-R (1 camada), ANN-R (2 camadas) e ANN WTA.

Oper.		ANN FF	ANN-R (1)	ANN-R (2)	ANN WTA
Pert. C_{Af}	DF/F	**0,99375**	0,98125	0,98125	0,96250
	DF/N	0,00417	0,22917	0,10000	0,10417
	DN/N	**0,99583**	0,77083	0,90000	0,89583
	DN/F	0,00625	0,01875	0,01875	0,03750
	DF_i/F_j	-	-	-	-
Falha C_B	DF/F	**1,00000**	0,99167	0,99583	0,94583
	DF/N	0,00000	0,00000	0,00000	0,00000
	DN/N	**1,00000**	**1,00000**	**1,00000**	**1,00000**
	DN/F	0,00000	0,00833	0,00417	0,05417
	DF_i/F_j	-	-	-	-
Falha T	DF/F	**1,00000**	0,99583	0,99583	0,96666
	DF/N	0,00000	0,00000	0,00000	0,00000
	DN/N	**1,00000**	**1,00000**	**1,00000**	**1,00000**
	DN/F	0,00000	0,00417	0,00417	0,01667
	DF_i/F_j	-	-	-	0,01667
Emp. F	DF/F	0,99583	**1,00000**	**1,00000**	**1,00000**
	DF/N	0,00000	0,00000	0,00000	0,00000
	DN/N	**1,00000**	**1,00000**	**1,00000**	**1,00000**
	DN/F	0,00417	0,00000	0,00000	0,00000
	DF_i/F_j	-	-	-	-

Entre os métodos de redes neuronais artificiais, pode-se destacar que o método de classificação de aprendizagem supervisionada com alimentação direta (ANN FF) apresentou resultados consistentes, utilizando informações a respeito da operação normal e de todas as falhas conhecidas. Quanto ao método de regressão (ANN-R), apesar da impossibilidade do mesmo identificar a falha, os resultados foram muito bons, conseguindo detectar a operação anormal em todos os casos. Entre as duas configurações de redes neuronais de regressão utilizadas, com 1 camada e com 2 camadas ocultas, a configuração que apresentou os melhores resultados foi a rede neuronal artificial de regressão com 2 camadas ocultas (configuração 10:10:10:2). Já o método de RNA com aprendizado não-supervisionado, a ANN *winner-take-all* (ANN WTA) conseguiu desempenhar a classificação de acordo com a proposta, porém sem a rapidez da ANN FF.

Entre os métodos ANN estudados, pode-se verificar que para detectar operações anormais (índice DF/F), o método de classificação de aprendizagem supervisionada com alimentação direta (ANN FF) foi o que teve melhor desempenho para todas as situações

avaliadas. Os métodos ANN-R também apresentaram desempenhos consideráveis na detecção de operação normal (índice DN/N), entretanto a detecção de operações anormais foi menor que da ANN FF. No caso em que o sistema de detecção de falhas também identifica a falha, em metodologias com redes neuronais artificiais de classificação supervisionadas com alimentação direta (ANN FF) e não-supervisionadas (ANN WTA), a ANN de classificação supervisionada obteve melhores resultados na detecção de operações normais, enquanto que na detecção de falhas, a ANN *winner-take-all* (ANN WTA) obteve um desempenho menor, porém consistente. Deve-se deixar claro que existiram erros de classificação somente com o modelo ANN WTA, na detecção da falha *T*. Avaliando as necessidades de cada método, o método de aprendizagem não-supervisionada se mostrou o mais competente, não necessitando o fornecimento de rótulos no treinamento, nem definição quanto a quantidade de épocas ou quanto a estrutura da rede, somente a quantidade de nós de saída ou classes dos dados a serem treinados.

5.10.6. Análise de Falhas com Diferentes Amplitudes

A aplicação de redes neuronais de classificação e de regressão para falhas com diferentes amplitudes das falhas treinadas aos modelos foram realizadas com os modelos treinados anteriormente, para ANN FF, ANN-R (2 camadas ocultas) e ANN WTA. Para melhor efeito de comparação, foram separados os resultados para cada uma das falhas propostas no capítulo 3.

5.10.6.1. Perturbação C_{Af}

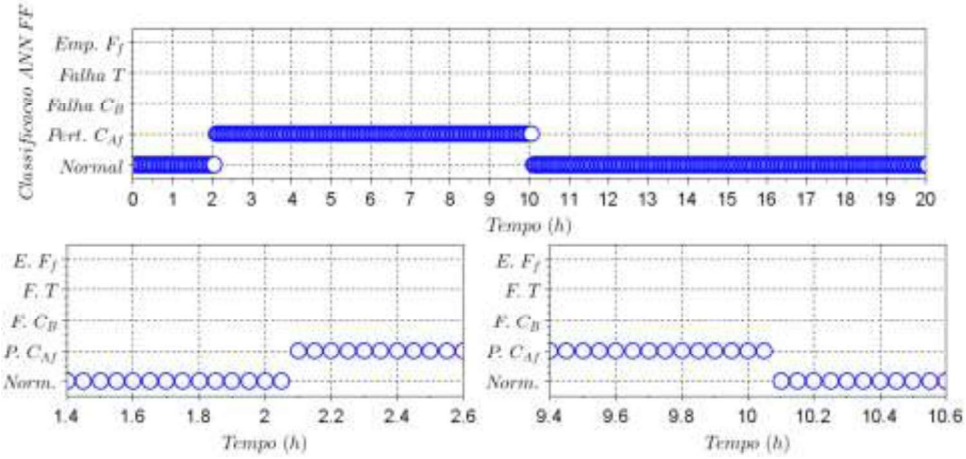
Na Seção 3.5.2, foram estabelecidas diferentes amplitudes das falhas pesquisadas e treinadas nos sistemas de detecção e diagnóstico de falhas. Para a perturbação C_{Af}, foram estabelecidos dois valores diferentes para C_{Af}, 4,5 mol/L e 4,9 mol/L.

A Figura 5.48 apresenta o resultado da detecção da perturbação C_{Af} pelo modelo de redes neuronais artificiais de alimentação direta (ANN FF) quando utilizado valor de C_{Af} menor (4,5 mol/L) que o valor de C_{Af} utilizado no treinamento (4,7 mol/L) para o método de detecção de falhas.

A perturbação C_{Af} menor (4,5 mol/L) foi detectada pelo modelo ANN FF mais rapidamente que a perturbação C_{Af} de 4,7 mol/L treinada no modelo ANN FF. Para os dados com amplitude diferente, o sistema de detecção levou 4 intervalos de amostragem (0,20 h) para detectar a falha, enquanto que para os dados com amplitude igual aos dados de treinamento, o sistema de detecção levou 7 intervalos de amostragem (0,35 h). Após a perturbação C_{Af} ter sido removida, o sistema de detecção levou 9 intervalos de amostragem a mais (0,45 h) para detectar o retorno a operação normal do que quando utilizados os dados com a mesma amplitude dos

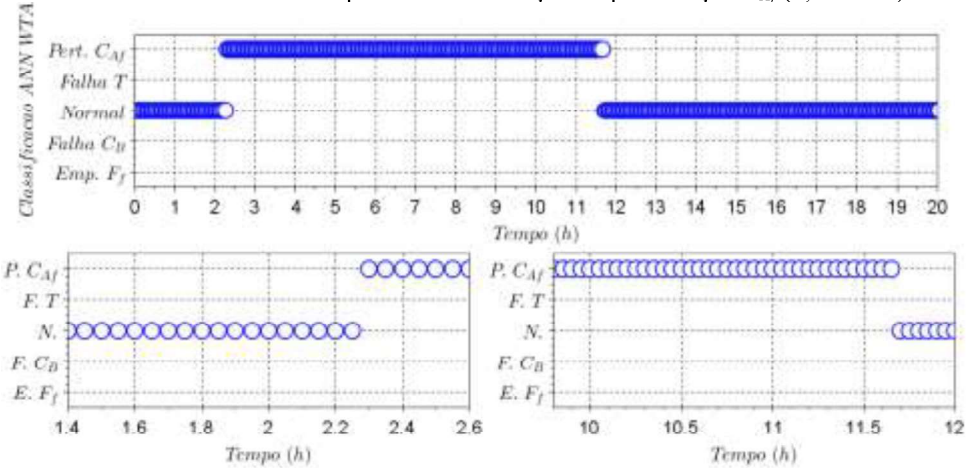
dados de treinamento (C_{Af} = 4,7 mol/L, 16 intervalos de amostragem, 0,8 h; C_{Af} = 4,5 mol/L, 25 intervalos de amostragem, 1,25 h).

Figura 5.48 - Rótulos obtidos através do processo de detecção da perturbação C_{Af} (4,5 mol/L) - ANN FF.



A Figura 5.49 apresenta o resultado da detecção da perturbação C_{Af} pelo modelo de rede neuronal artificial não supervisionada *winner-take-all* (ANN WTA) quando utilizado valor de C_{Af} menor (4,5 mol/L) que o valor de C_{Af} utilizado no treinamento (4,7 mol/L) para o método de detecção de falhas.

Figura 5.49 - Rótulos obtidos através do processo de detecção da perturbação C_{Af} (4,5 mol/L) - ANN WTA.

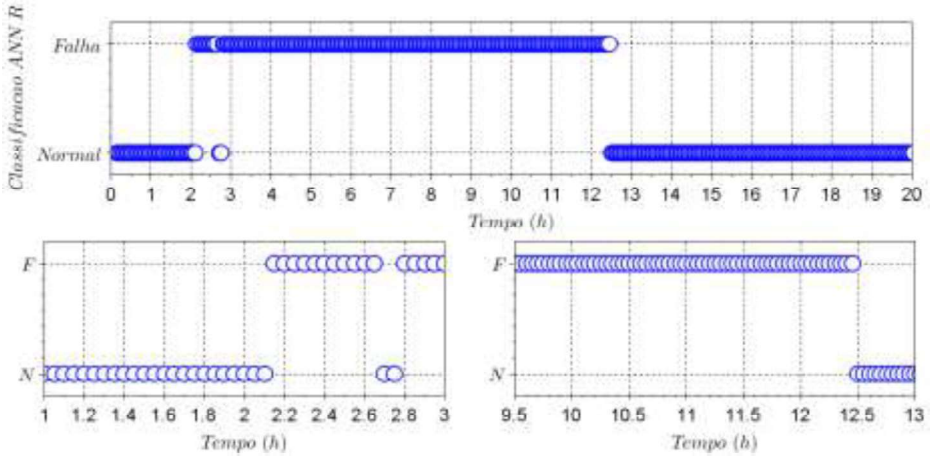


A perturbação C_{Af} menor (4,5 mol/L) foi detectada pelo modelo ANN WTA mais rapidamente que a perturbação C_{Af} de 4,7 mol/L pelo mesmo modelo. Para os dados com amplitude diferente, o sistema de detecção levou 5 intervalos de amostragem (0,25 h) para detectar a falha, enquanto que para os dados com amplitude igual aos dados de treinamento, o sistema de detecção levou 6 intervalos de amostragem (0,30 h). Após a perturbação C_{Af} ter sido removida, o sistema de detecção levou 8 intervalos de amostragem a mais (0,40 h) para detectar o retorno a operação normal do que quando utilizados os dados com a mesma amplitude dos

dados de treinamento (C_{Af} = 4,7 mol/L, 25 intervalos de amostragem, 1,25 h; C_{Af} = 4,5 mol/L, 33 intervalos de amostragem, 1,65 h).

A Figura 5.50 apresenta o resultado da detecção da perturbação C_{Af}, pelo modelo de rede neuronal artificial de regressão (ANN-R) quando utilizado valor de C_{Af} de 4,5 mol/L nos dados a serem detectados.

Figura 5.50 - Rótulos obtidos através do processo de detecção da perturbação C_{Af} (4,5 mol/L) - ANN-R.



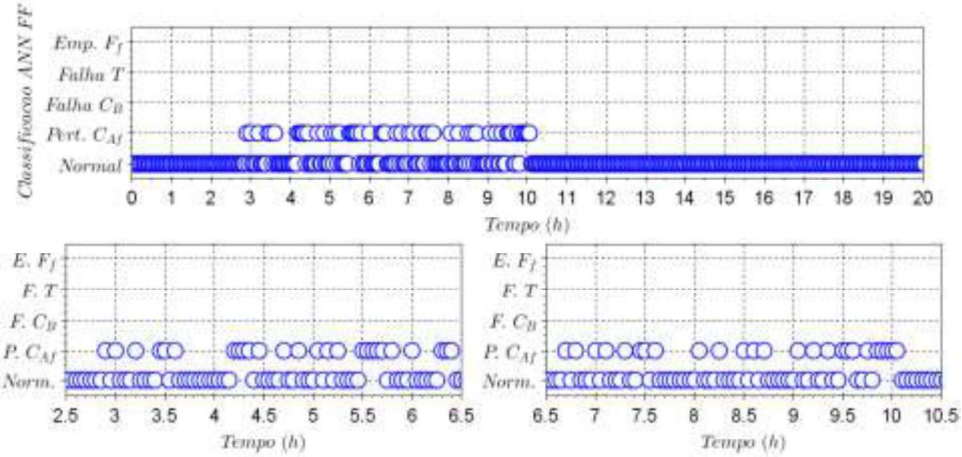
A perturbação C_{Af} menor (4,5 mol/L) foi detectada pelo modelo ANN-R após 2 intervalos de amostragem (0,10 h), porém, após 11 intervalos de amostragem (0,55 h) detectando a falha, o sistema detectou incorretamente a operação normal durante dois intervalos de amostragem (0,10 h). Após o instante 2,75 h, o sistema de detecção passou a somente detectar a operação anormal. Uma vez que a perturbação C_{Af} foi removida no instante 10,0 h, o sistema de detecção passou 49 intervalos de amostragem (2,45 h) para detectar o retorno a operação normal (C_{Af} = 4,7 mol/L, 24 intervalos de amostragem, 1,2 h; C_{Af} = 4,5 mol/L, 49 intervalos de amostragem, 2,45 h).

A Figura 5.51 apresenta o resultado da detecção da perturbação C_{Af} pelo modelo de redes neuronais artificiais de classificação com alimentação direta (ANN FF) quando utilizado valor de C_{Af} maior (4,9 mol/L) que o valor de C_{Af} utilizado no treinamento (4,7 mol/L) do método de detecção de falhas.

A perturbação C_{Af} maior (4,9 mol/L) foi detectada de maneira incorreta pelo modelo ANN FF que a perturbação C_{Af} de 4,7 mol/L treinada no mesmo modelo. Para os dados com amplitude diferente, o sistema de detecção conseguiu detectar somente alguns intervalos de amostragem ao longo de todo o período em que se manteve a perturbação C_{Af} presente. Ao longo do intervalo de 2 h até 10 h, a falha foi detectada somente em 50 intervalos de amostragem desencontrados, enquanto que o sistema, no restante do período, identificou que o processo continuava em operação normal (110 intervalos de amostragem). A detecção foi extremamente

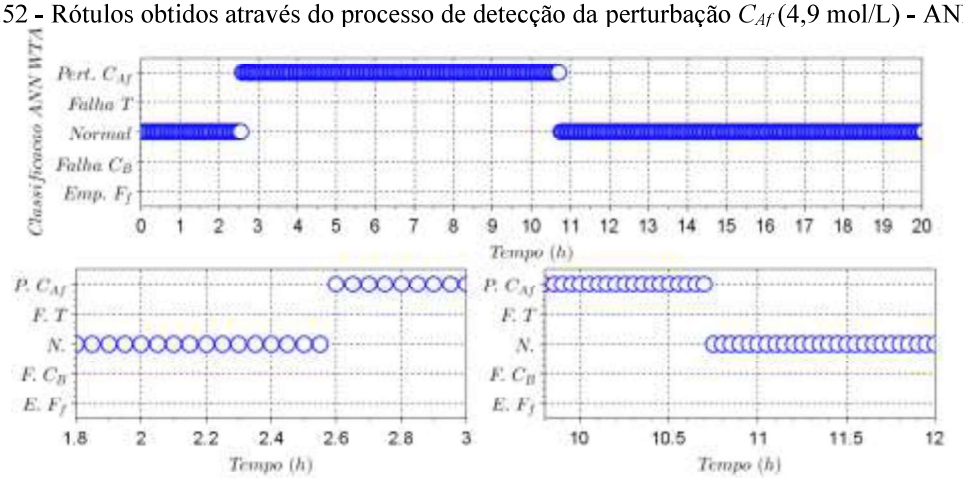
ruim para o método de rede neuronal artificial de classificação com alimentação direta, quando a amplitude da falha perturbação C_{Af} era maior que a amplitude da falha utilizada no treinamento do modelo.

Figura 5.51 - Rótulos obtidos através do processo de detecção da perturbação C_{Af} (4,9 mol/L) - ANN FF.



A Figura 5.52 apresenta o resultado da detecção da perturbação C_{Af} pelo modelo redes neuronais artificiais não supervisionadas *winner-take-all* (ANN WTA) quando utilizado valor de C_{Af} maior (4,9 mol/L) que o valor de C_{Af} utilizado no treinamento (4,7 mol/L) do método ANN WTA de detecção de falhas.

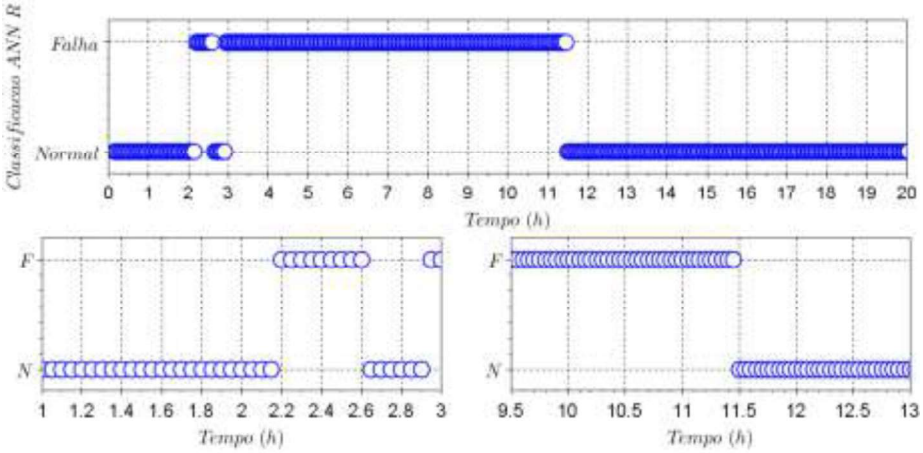
Figura 5.52 - Rótulos obtidos através do processo de detecção da perturbação C_{Af} (4,9 mol/L) - ANN WTA.



A perturbação C_{Af} maior (4,9 mol/L) foi detectada pelo modelo ANN FF a partir do instante 2,55 h (após 0,55 h do início da falha). O método foi capaz de detectar a falha 5 intervalos de amostragem mais lentamente que os dados utilizados no treinamento do modelo (C_{Af} = 4,7 mol/L, 6 intervalos de amostragem, 0,30 h; C_{Af} = 4,9 mol/L, 11 intervalos de amostragem, 0,55 h). Após a finalização da perturbação no instante 10 h, o sistema de detecção identificou a operação normal após 14 intervalos de amostragem (0,70 h).

A Figura 5.53 apresenta o resultado da detecção dos dados da perturbação C_{Af} pelo modelo de redes neuronais artificiais de regressão quando utilizado valor de C_{Af} de 4,9 mol/L.

Figura 5.53 - Rótulos obtidos através do processo de detecção da perturbação C_{Af} (4,9 mol/L) - ANN-R.



A perturbação C_{Af} maior (4,9 mol/L) foi detectada pelo modelo ANN-R após 3 intervalos de amostragem (0,15 h), porém, após 9 intervalos de amostragem (0,45 h) detectando a falha, o sistema detectou incorretamente a operação normal durante 6 intervalos de amostragem (0,30 h). Após o instante 2,90 h, o sistema de detecção passou a somente detectar a operação anormal. Uma vez que a perturbação C_{Af} foi removida no instante 10,0 h, o sistema de detecção passou 29 intervalos de amostragem (1,45 h) para detectar o retorno a operação normal (C_{Af} = 4,7 mol/L, 24 intervalos de amostragem, 1,2 h; C_{Af} = 4,9 mol/L, 29 intervalos de amostragem, 1,45 h).

A Tabela 5.7 apresenta os instantes de detecção para as diferentes metodologias e diferentes amplitudes da perturbação C_{Af}, assim como os índices obtidos para cada situação.

Tabela 5.7 - Índices - ANN FF, ANN-R e ANN WTA - amplitudes perturbação C_{Af}.

C_{Af}	Metodologia	TAD (h)	DF/F	DF/N	DN/N	DN/F
4,5 mol/L	ANN FF	**2,10**	**0,99375**	0,00417	**0,99583**	0,00625
	ANN WTA	2,30	0,96875	0,13750	0,86250	0,03125
	ANN-R	2,15	0,97500	0,20417	0,79583	0,02500
4,7 mol/L	ANN FF	**2,10**	**0,99375**	0,00417	**0,99583**	0,00625
	ANN WTA	2,35	0,96250	0,10417	0,89583	0,03750
	ANN-R	2,20	0,98125	0,10000	0,90000	0,01875
4,9 mol/L	ANN FF	-	0,31250	0,00417	**0,99583**	0,68750
	ANN WTA	2,60	0,93125	0,05833	0,94167	0,06875
	ANN-R	**2,20**	**0,94375**	0,12083	0,87917	0,05625

Para as diferentes amplitudes da perturbação C_{Af} todos os modelos reconheceram valores maiores e valores menores de C_{Af}. A metodologia ANN de classificação com algoritmo de alimentação direta (ANN FF) não conseguiu detectar de modo satisfatório a perturbação C_{Af}

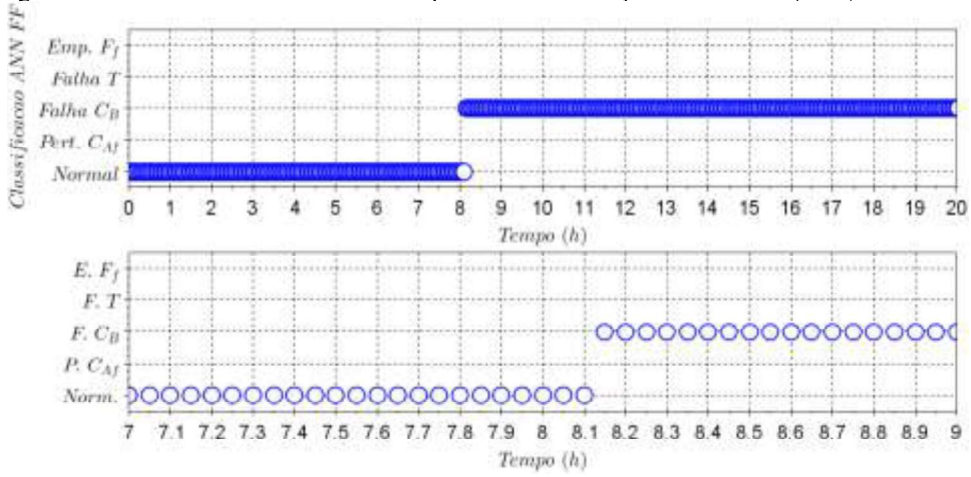
com o valor de C_{Af} igual a 4,9 mol/L. Neste caso, como a variação de C_{Af} (5,1 – 4,9 = 0,2 mol/L) foi muito pequena, o método não foi capaz de diferenciar da operação normal os dados com a perturbação C_{Af}. Entre as diferenças dos métodos, o ANN-R foi o único capaz de identificar de maneira consistente em todas as três situações com diferentes C_{Af}. Já o método ANN com aprendizado não supervisionado (ANN WTA) apresentou dados de detecção com grandes atrasos da operação anormal, assim como o retorno a operação normal.

5.10.6.2. Falha C_B

Na Seção 3.5.2, foram estabelecidas diferentes amplitudes das falhas pesquisadas e treinadas nos sistemas de detecção e diagnóstico de falhas. Para a falha C_B, foram estabelecidos aumentos abruptos menor (10%) e maior (30%) no valor medido da variável controlada C_B após a aplicação da falha, diferentes do aumento abrupto utilizado para treinamento (20%).

A Figura 5.54 apresenta o resultado da detecção da falha C_B pela ANN de classificação de alimentação direta (ANN FF) quando utilizado aumento abrupto menor (10% do valor medido de C_B) que o aumento abrupto utilizado no treinamento (20%) do método de detecção de falhas.

Figura 5.54 - Rótulos obtidos através do processo de detecção da falha C_B (10%) - ANN FF.

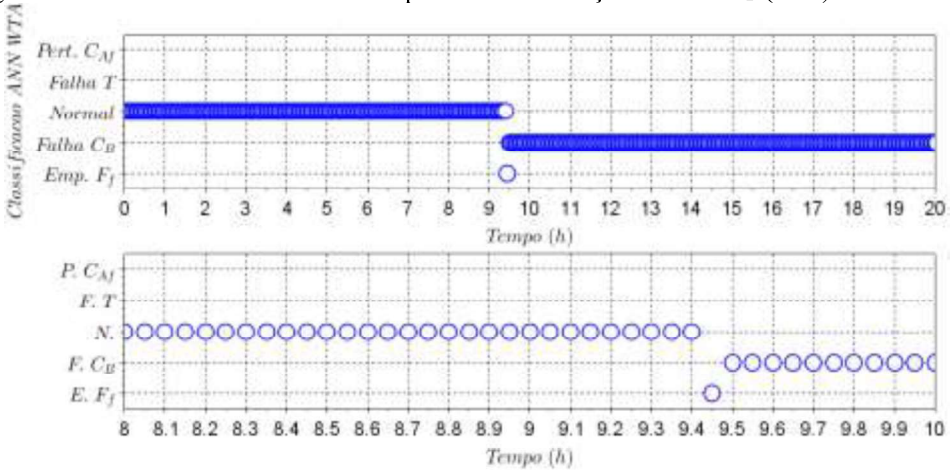


A falha C_B obtida através do aumento abrupto menor (10%) foi detectada pelo modelo ANN FF mais lentamente que a falha C_B obtida com o aumento abrupto de 20% da variável medida. Para os dados com amplitude diferente, o sistema de detecção levou 2 intervalos de amostragem, ou 0,10 h, para detectar a falha, enquanto que para os dados com amplitude igual aos dados de treinamento, o sistema de detecção identificou instantaneamente a falha, no instante seguinte a aplicação da falha em C_B.

A Figura 5.55 apresenta o resultado da detecção da falha C_B pelo modelo ANN de classificação com aprendizado não supervisionado *winner-take-all* (ANN WTA) quando

utilizado aumento abrupto menor (10% do valor medido de C_B) que o aumento abrupto utilizado no treinamento (20%) pelo mesmo método de detecção de falhas.

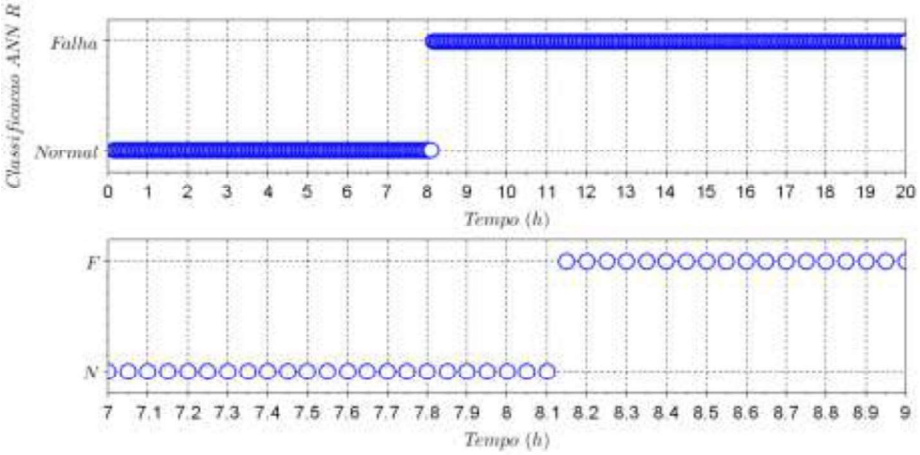
Figura 5.55 - Rótulos obtidos através do processo de detecção da falha C_B (10%) - ANN WTA.



A falha C_B obtida através do aumento abrupto menor (10%) foi detectada pelo modelo ANN WTA mais lentamente que a falha C_B obtida com o aumento abrupto de 20% da variável medida. Para os dados com amplitude diferente, o sistema de detecção levou 30 intervalos de amostragem (1,50 h) para detectar a falha, enquanto que para os dados com amplitude igual aos dados de treinamento, o sistema de detecção levou 13 intervalos de amostragem (0,65 h). O ANN WTA cometeu um pequeno erro de classificação no instante 9,45 h, quando identificou a presença da operação anormal emperramento F_f durante um intervalo de amostragem.

A Figura 5.56 apresenta o resultado da detecção da falha C_B pelo modelo de rede neuronal artificial de regressão quando utilizado aumento abrupto menor (10% do valor medido de C_B) que o aumento abrupto apresentado na Seção 5.10.3 (20% do valor medido de C_B).

Figura 5.56 - Rótulos obtidos através do processo de detecção da falha C_B (10%) - ANN-R.

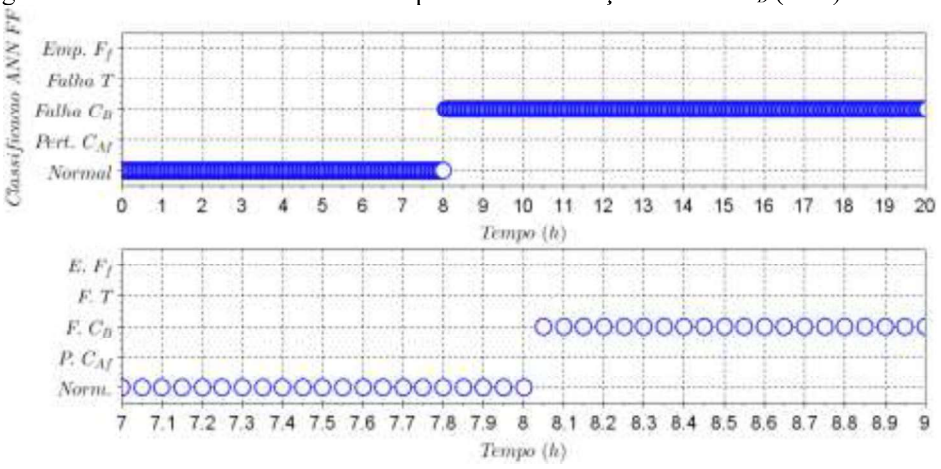


A falha C_B obtida através do aumento abrupto de 10% da variável C_B foi detectada pelo

modelo ANN-R após 2 intervalos de amostragem (0,10 h), enquanto que nos dados com aumento abrupto de 20% da variável C_B para a ANN-R, a falha C_B foi detectada após 1 intervalo de amostragem (0,05 h).

A Figura 5.57 apresenta o resultado da detecção da falha C_B utilizando a metodologia de redes neuronais artificiais de classificação com alimentação direta (ANN FF) obtida através dos dados com aumento abrupto de 30% da variável controlada C_B.

Figura 5.57 - Rótulos obtidos através do processo de detecção da falha C_B (30%) - ANN FF.

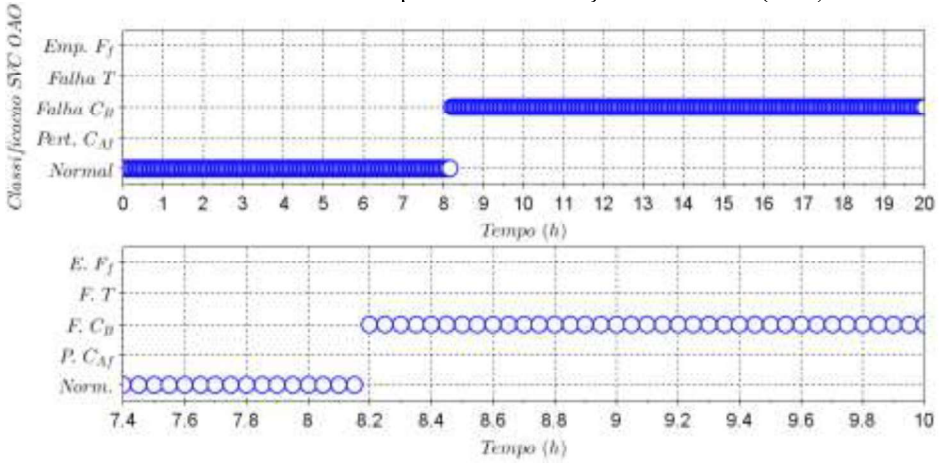


A falha C_B obtida através do maior aumento abrupto da variável controlada (30%) foi detectada pelo modelo ANN FF da mesma maneira que a falha C_B obtida através do aumento abrupto de 20%. Para os dados com amplitude diferente e com mesma amplitude dos dados de treinamento, a detecção da falha ocorreu instantaneamente, no exato instante após a aplicação da falha (8,00 h).

A Figura 5.58 apresenta o resultado da detecção da falha C_B utilizando a metodologia de redes neuronais artificiais de classificação com aprendizado não supervisionado *winner-take-all* (ANN WTA) obtido através dos dados com aumento abrupto de 30% da variável controlada C_B.

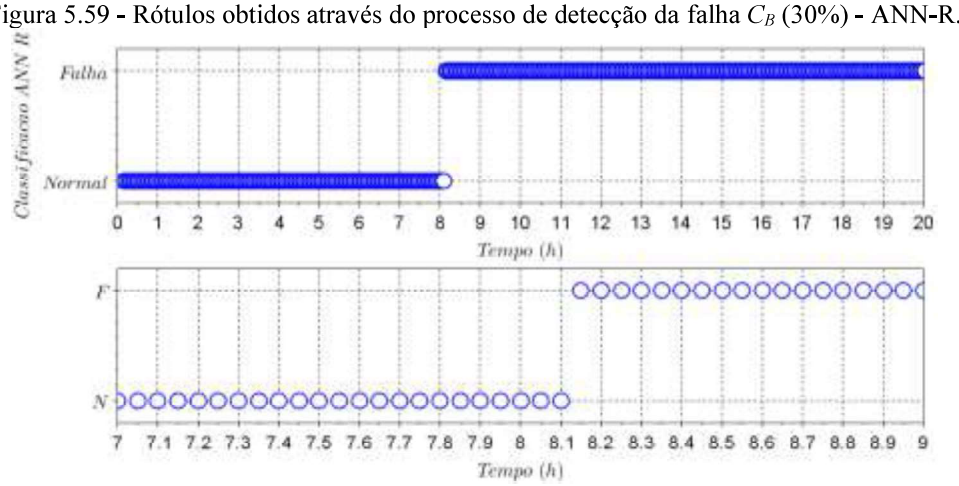
A falha C_B obtida através do maior aumento abrupto da variável controlada (30%) foi detectada pelo modelo ANN WTA mais rapidamente que a falha C_B obtida através do aumento abrupto de 20%. Para os dados com amplitude diferente, o sistema de detecção identificou a falha corretamente após 8 intervalos de amostragem (0,40 h), enquanto que para os dados com amplitude igual aos dados de treinamento, o sistema de detecção identificou a falha após 13 intervalos de amostragem (0,35 h).

Figura 5.58 - Rótulos obtidos através do processo de detecção da falha C_B (30%) - ANN WTA.



A Figura 5.59 apresenta o resultado da detecção da falha C_B com a utilização da rede neuronal artificial de regressão (ANN-R) quando utilizado aumento abrupto maior (30% do valor medido de C_B) que o aumento abrupto apresentado na Seção 5.10.3 (20% do valor medido de C_B).

Figura 5.59 - Rótulos obtidos através do processo de detecção da falha C_B (30%) - ANN-R.



A falha C_B obtida através do aumento abrupto de 30% da variável C_B foi detectada pelo modelo ANN-R após 2 intervalos de amostragem (0,10 h), enquanto que nos dados com aumento abrupto de 20% da variável C_B para a ANN-R, a falha C_B foi detectada após 1 intervalo de amostragem (0,05 h).

A Tabela 5.8 apresenta os instantes de detecção para as diferentes metodologias de detecção de falhas utilizando redes neuronais artificias (TAD) e diferentes amplitudes da falha C_B, assim como os índices obtidos para cada situação.

Todos os modelos reconheceram as diferentes amplitudes da falha C_B. Entre as metodologias ANN de classificação, a ANN *Feedforward* com aprendizado supervisionado detectou a falha C_B mais rapidamente que a metodologia ANN *winner-take-all*. Entre as

diferentes metodologias, a ANN FF e a ANN-R foram os modelos capazes de identificar de maneira consistente em todas as três situações com diferentes C_B. Ambas as metodologias detectaram todas as falhas em até 2 intervalos de amostragem (0,10 h). Comparando as duas metodologias, a vantagem da ANN FF está justamente na capacidade de identificar a falha em questão enquanto que a ANN-R somente indica se o sistema passa ou não por uma situação anormal. Entre os três métodos, o único que apresentou um erro de identificação da falha em questão foi a rede neuronal artificial de classificação *winner-take-all* (ANN WTA).

Tabela 5.8 - Índices - ANN FF, ANN WTA e ANN-R - amplitudes falha C_B.

C_B	Metodologia	TAD (h)	DF/F	DF/N	DN/N	DN/F	DF_i/F_j
10%	ANN FF	**8,15**	**0,99167**	0,00000	1,00000	0,00833	-
	ANN WTA	9,50	0,87917	0,00000	1,00000	0,11667	0,00417
	ANN-R	**8,15**	**0,99167**	0,00000	1,00000	0,00833	-
20%	ANN FF	**8,05**	**1,00000**	0,00000	1,00000	0,00000	-
	ANN WTA	8,70	0,94583	0,00000	1,00000	0,05417	-
	ANN-R	8,10	0,99583	0,00000	1,00000	0,00417	-
30%	ANN FF	**8,05**	**1,00000**	0,00000	1,00000	0,00000	-
	ANN WTA	8,45	0,96667	0,00000	1,00000	0,03333	-
	ANN-R	8,15	0,99167	0,00000	1,00000	0,00833	-

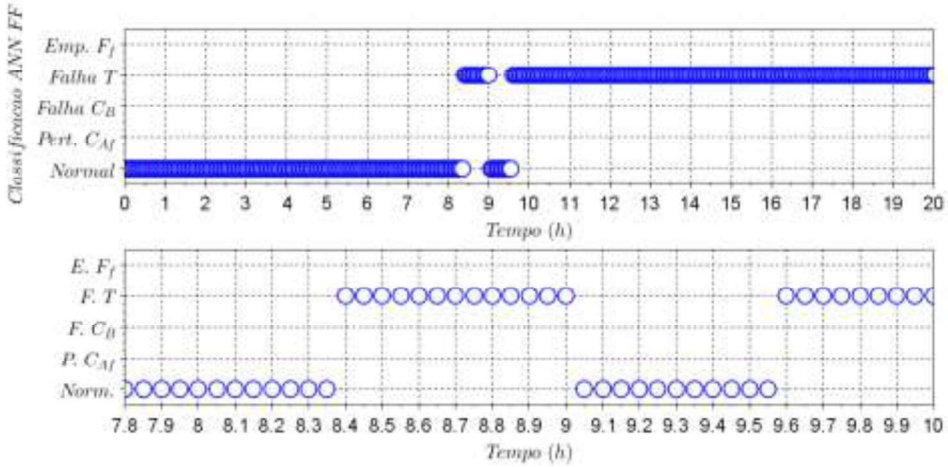
5.10.6.3. Falha *T*

Na Seção 3.5.2, foram estabelecidas diferentes amplitudes das falhas pesquisadas e treinadas nos sistemas de detecção e diagnóstico de falhas baseados em redes neuronais artificiais. Para a falha *T*, foram estabelecidas medidas incorretas do sensor de temperatura do reator (0,5% e 2,0% maiores que a medida imediatamente anterior a aplicação da falha) da variável controlada *T*, diferentes da medida incorreta do sensor de temperatura do reator utilizada para treinamento (1,0% maior).

A Figura 5.60 apresenta o resultado da detecção da falha *T* pela metodologia de redes neuronais artificiais supervisionadas de classificação com alimentação direta (ANN FF) quando utilizada medida incorreta do sensor de temperatura menor (0,5% maior que a medida de *T*) que a medida incorreta utilizada no treinamento (1,0% maior) do método de detecção de falhas.

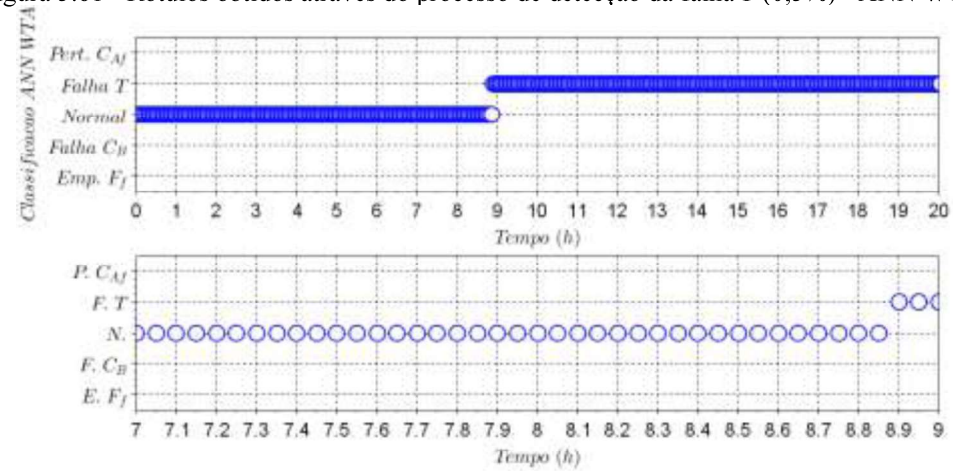
A falha *T* obtida através da medida incorreta do sensor *T* menor (0,5%) foi detectada pelo modelo ANN FF após 7 intervalos de amostragem (0,35 h), porém devido a dinâmica dos sinais manipulados e controlados, após 13 intervalos de amostragem (0,65 h) o sistema de detecção voltou a detectar a operação normal durante 11 intervalos de amostragem (0,55 h) até o instante 9,55 h. Após os 11 intervalos de amostragem, o sistema voltou a detectar a falha *T*.

Figura 5.60 - Rótulos obtidos através do processo de detecção da falha *T* (0,5%) - ANN FF.



A Figura 5.61 apresenta o resultado da detecção da falha *T* pelo modelo de rede neuronal artificial de classificação com aprendizado não supervisionado *winner-take-all* (ANN WTA) quando utilizada medida incorreta do sensor de temperatura menor (0,5% maior que a medida de *T*) que a medida incorreta utilizada no treinamento (1,0% maior) do método de detecção de falhas.

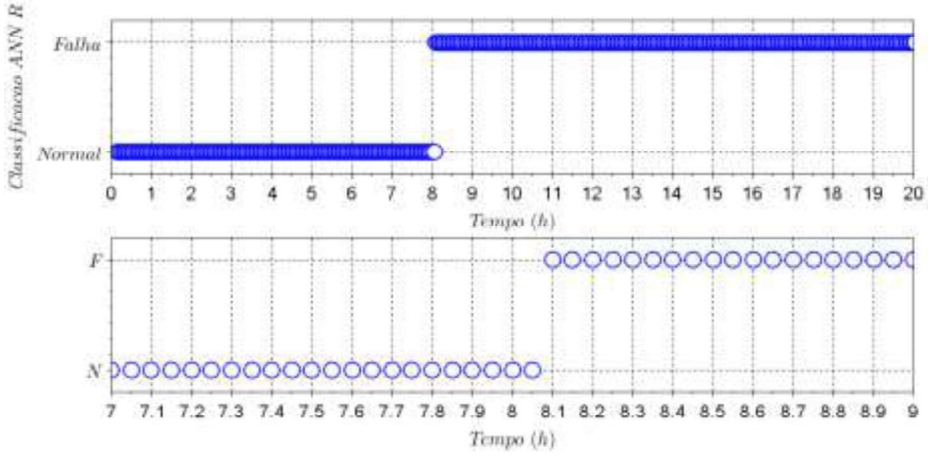
Figura 5.61 - Rótulos obtidos através do processo de detecção da falha *T* (0,5%) - ANN WTA.



A falha *T* obtida através da medida incorreta do sensor *T* menor (0,5% maior que a medida incorreta anterior a aplicação da falha) foi detectada pelo modelo ANN WTA após 17 intervalos de amostragem (0,85 h), mais lentamente que a medida incorreta do sensor *T* utilizada no treinamento (1,0% maior), detectada instantaneamente apesar do comportamento apresentado.

A Figura 5.62 apresenta o resultado da detecção da falha *T* pelo modelo de rede neuronal artificial de regressão quando a medida incorreta é menor (0,5% maior que a medida incorreta do sensor de temperatura *T*) que a medida incorreta apresentada na Seção 5.10.3 (1,0% maior).

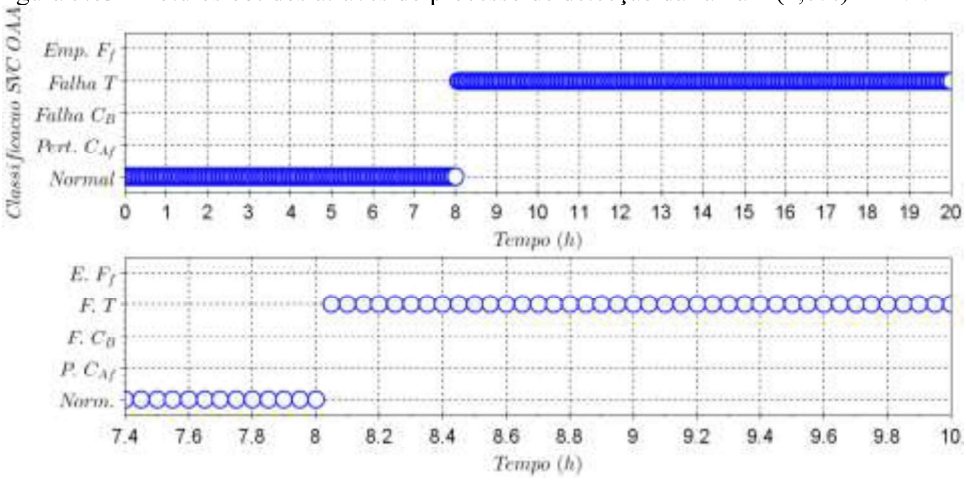
Figura 5.62 - Rótulos obtidos através do processo de detecção da falha *T* (0,5%) - ANN-R.



A falha *T* obtida através da medida incorreta do sensor de temperatura *T* do reator 0,5% maior que a medida anterior a aplicação da falha *T* foi detectada pelo modelo ANN-R dentro da mesma quantidade de intervalos de amostragem que os dados com medida incorreta 1,0% maior da variável *T*. Para ambos os casos, o sistema de detecção levou 1 intervalo de amostragem (0,05 h) para detectar falha.

A Figura 5.63 apresenta o resultado da detecção da falha *T* pelo modelo de rede neuronal artificial de classificação com alimentação direta (ANN FF) quando utilizada medida incorreta do sensor de temperatura maior (2,0% maior que a medida de *T*) que a medida incorreta utilizada no treinamento (1,0% maior) do método de detecção de falhas.

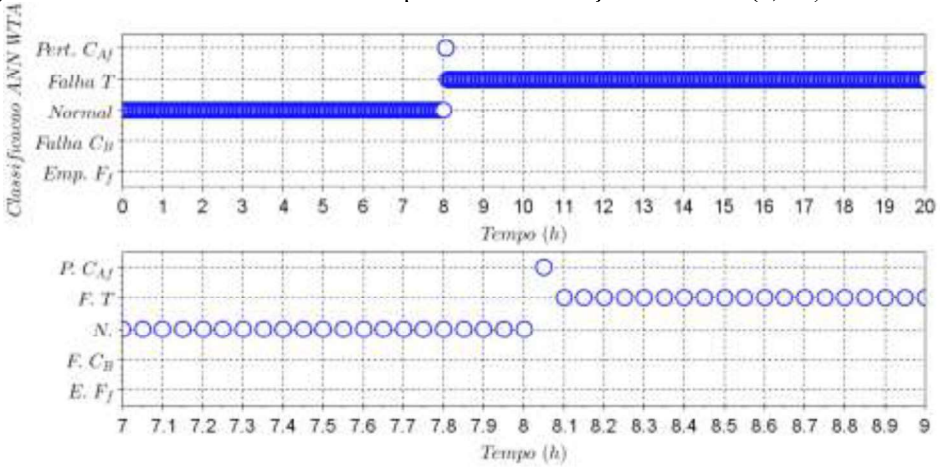
Figura 5.63 - Rótulos obtidos através do processo de detecção da falha *T* (2,0%) - ANN FF.



A falha *T* obtida através da medida incorreta do sensor *T* maior (2,0%) foi detectada pelo modelo ANN FF instantaneamente, de maneira idêntica a forma que o modelo detectou a falha *T* com dados obtidos através da medida incorreta do sensor *T* 1,0% maior que a última medida correta.

A Figura 5.64 apresenta o resultado da detecção da falha *T* obtido através da rede neuronal artificial de classificação não supervisionada *winner-take-all* (ANN WTA) quando utilizada medida incorreta do sensor de temperatura maior (2,0% maior que a medida de *T*) que a medida incorreta utilizada no treinamento (1,0% maior) do método de detecção de falhas.

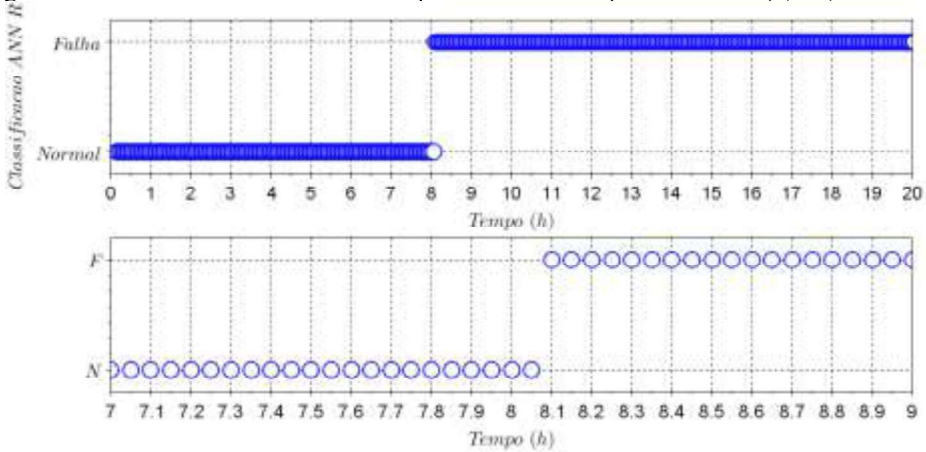
Figura 5.64 - Rótulos obtidos através do processo de detecção da falha *T* (2,0%) - ANN WTA.



A falha *T* obtida através da medida incorreta do sensor *T* maior (2,0% maior que a medida incorreta anterior a aplicação da falha) foi detectada pelo modelo ANN WTA após o instante 8,10 h. No instante 8,05 h, a ANN WTA identificou erroneamente a perturbação C_{Af} durante um intervalo de amostragem (0,05 h). A rede neuronal *Winner-take-all* obteve um melhor comportamento de detecção nesta situação do que na situação de identificar os dados da falha no sensor *T* com a amplitude utilizada no treinamento (1,0% maior).

A Figura 5.65 apresenta o resultado da detecção da falha *T* pelo modelo de rede neuronal artificial de regressão quando a medida incorreta é maior (2,0% maior que a medida incorreta do sensor de temperatura *T*) que a medida incorreta apresentada na Seção 5.10.3 (1,0% maior).

Figura 5.65 - Rótulos obtidos através do processo de detecção da falha *T* (2,0%) - ANN-R.



A falha *T* obtida através da medida incorreta do sensor de temperatura *T* do reator 2,0% maior que a medida anterior a aplicação da falha *T* foi detectada pelo modelo ANN-R dentro da mesma quantidade de intervalos de amostragem que os dados com medida incorreta 1,0% maior da variável *T*. Para ambos os casos, o sistema de detecção levou 1 intervalo de amostragem (0,05 h) para detectar a operação anormal.

A Tabela 5.9 apresenta os instantes de detecção para as diferentes metodologias e diferentes amplitudes da falha *T*, assim como os índices obtidos para cada situação.

Tabela 5.9 - Índices - ANN FF, ANN WTA e ANN-R - amplitudes falha *T*.

T	Metodologia	TAD (h)	DF/F	DF/N	DN/N	DN/F	DF_i/F_j
0,5%	ANN FF	8,40	0,92500	0,00000	1,00000	0,07500	-
	ANN WTA	8,90	0,92917	0,00000	1,00000	0,07083	-
	ANN-R	**8,10**	**0,99583**	0,00000	1,00000	0,00417	-
1%	ANN FF	**8,05**	**1,00000**	0,00000	1,00000	0,00000	-
	ANN WTA	8,45	0,96666	0,00000	1,00000	0,01667	0,01667
	ANN-R	8,10	0,99583	0,00000	1,00000	0,00417	-
2%	ANN FF	**8,05**	**1,00000**	0,00000	1,00000	0,00000	-
	ANN WTA	8,10	0,99583	0,00000	1,00000	0,00000	0,00417
	ANN-R	8,10	0,99583	0,00000	1,00000	0,00417	-

Todos os modelos reconheceram as diferentes amplitudes da falha *T*. Entre as metodologias ANN de classificação, a ANN FF detectou a falha *T* mais rapidamente que o modelo ANN WTA, porém o modelo ANN WTA obteve algumas detecções incorretas, tanto para a falha *T* com medida incorreta 2,0% maior que a última medida válida, quanto para a falha *T* com medida incorreta 1,0% maior que a última medida válida. Entre as diferentes metodologias, a ANN-R foi a única capaz de detectar de maneira consistente em todas as três situações com diferentes *T* visualizando a falha após um intervalo de amostragem. Apesar de em duas situações (1,0% e 2,0%) a ANN FF ser o método que detecta e identifica mais rapidamente a falha (instantaneamente nos dois casos), a ANN-R apresenta comportamento mais robusto, conseguindo em todos os casos detectar a falha após um intervalo de amostragem (0,05 h). A única desvantagem do ANN-R é a incapacidade da identificação da operação anormal.

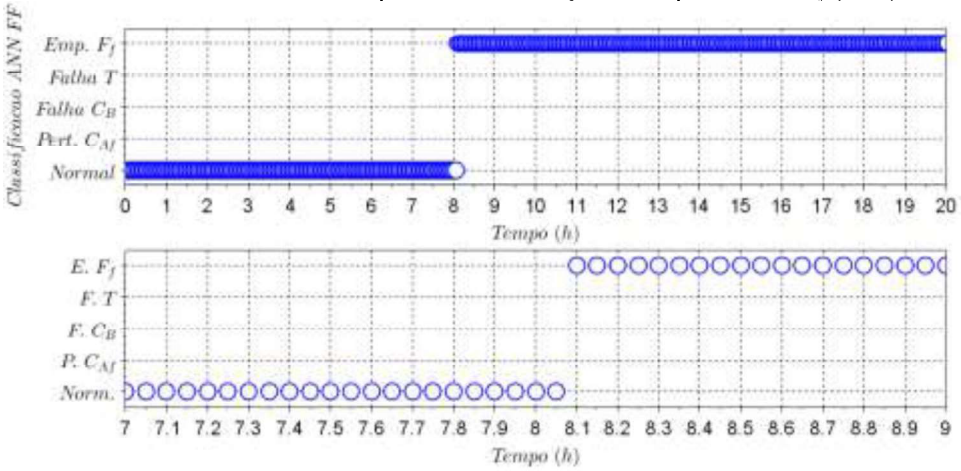
5.10.6.4. Emperramento F_f

Na Seção 3.5.2, foram estabelecidas diferentes amplitudes das falhas pesquisadas e treinadas nos sistemas de detecção e diagnóstico de falhas utilizando redes neuronais artificiais. Para o emperramento F_f, foram estabelecidos dados cuja vazão foi estabelecida a partir do momento em que foi aplicada a falha em 60% (menor) e 80% (maior) da vazão de alimentação do reator em estado estacionário, diferentes da vazão utilizada para os dados do treinamento

dos métodos do emperramento F_f (70% da vazão de alimentação do reator em estado estacionário).

A Figura 5.66 apresenta o resultado da detecção do emperramento F_f através da rede neuronal artificial supervisionada de classificação com algoritmo de alimentação direta (ANN FF) quando utilizada 60% da vazão de estado estacionário, vazão menor que a vazão de alimentação do reator utilizada no treinamento (70% da vazão de estado estacionário) do método de detecção de falhas.

Figura 5.66 - Rótulos obtidos através do processo de detecção do emperramento F_f (60%) - ANN FF.

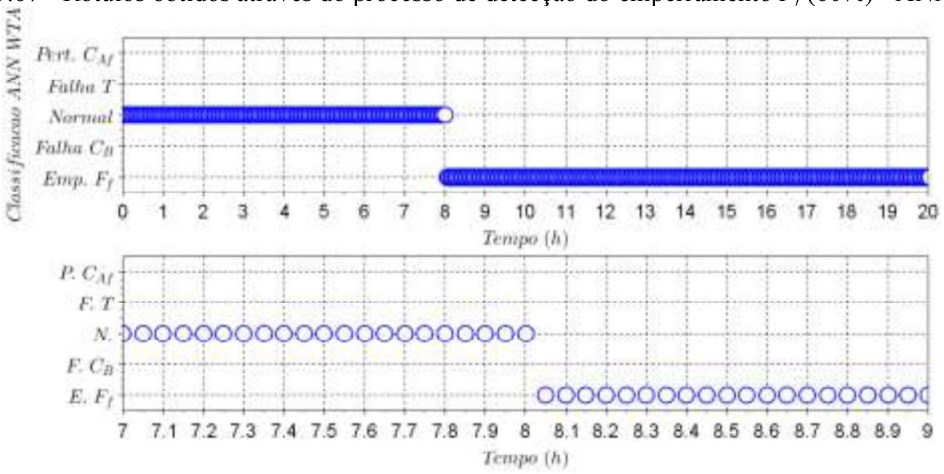


O emperramento F_f obtido através da modificação da vazão de alimentação do reator devido a um emperramento da válvula, em que a vazão passa a ser 60% da vazão de estado estacionário, foi detectada pelo modelo ANN FF após um intervalo de amostragem (0,05 h).

A Figura 5.67 apresenta o resultado da detecção do emperramento F_f com o uso da rede neuronal artificial de aprendizado não supervisionado *winner-take-all* (ANN WTA) quando utilizada 60% da vazão de alimentação do reator no estado estacionário, vazão menor que vazão utilizada no treinamento (70% da vazão no estado estacionário) do método de detecção de falhas.

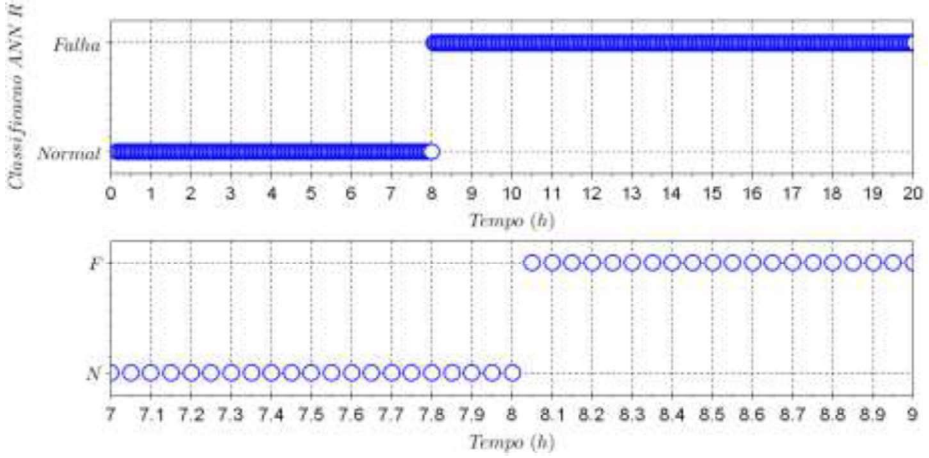
O emperramento F_f obtido através da modificação da vazão de alimentação do reator devido a um emperramento da válvula, em que a vazão passa a ser 60% da vazão de estado estacionário, foi detectada pelo modelo ANN WTA instantaneamente, de forma idêntica que o sistema detectou o emperramento F_f utilizado no treinamento (70% da vazão de alimentação do reator no estado estacionário).

Figura 5.67 - Rótulos obtidos através do processo de detecção do emperramento F_f (60%) - ANN WTA.



A Figura 5.68 apresenta o resultado da detecção do emperramento F_f pelo modelo de rede neuronal artificial de regressão quando a vazão de alimentação é menor (60% da vazão de alimentação do reator no estado estacionário) que a vazão de alimentação apresentada na Seção 5.10.3 (70% da vazão no estado estacionário).

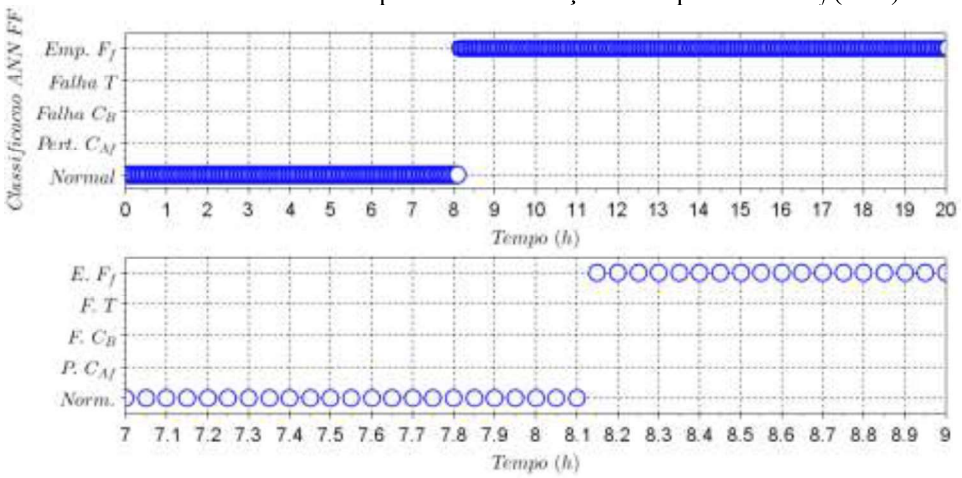
Figura 5.68 - Rótulos obtidos através do processo de detecção do emperramento F_f (60%) - ANN-R.



O emperramento F_f obtido através de 60% da vazão de alimentação do reator no estado estacionário foi detectado pelo modelo ANN-R dentro da mesma quantidade de intervalos de amostragem que os dados com 70% da vazão em estado estacionário. Para ambos os casos, a detecção ocorreu instantaneamente.

A Figura 5.69 apresenta o resultado da detecção do emperramento F_f pela metodologia de redes neuronais artificiais de classificação com algoritmo de alimentação direta (ANN FF) quando utilizada 80% da vazão de estado estacionário, vazão maior que a vazão de alimentação do reator utilizada no treinamento (70% da vazão de estado estacionário) do método de detecção de falhas.

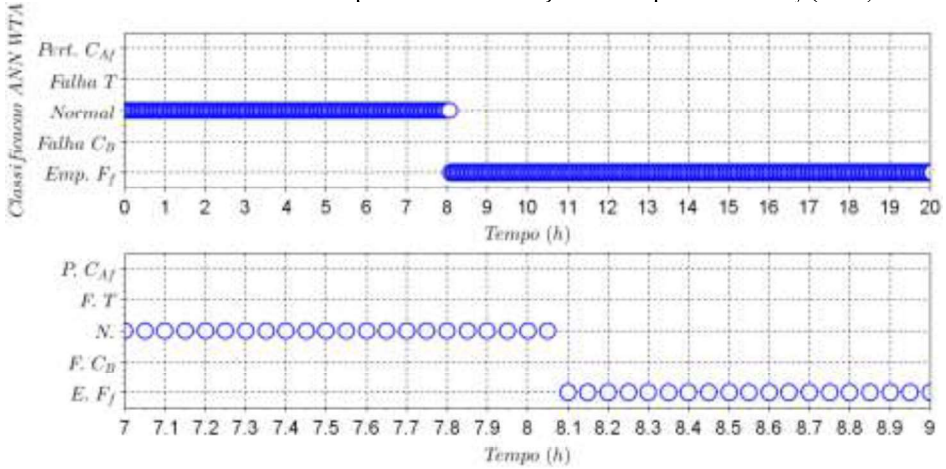
Figura 5.69 - Rótulos obtidos através do processo de detecção do emperramento F_f (80%) - ANN FF.



O emperramento F_f obtido através da modificação da vazão de alimentação do reator devido a um emperramento da válvula, em que a vazão passa a ser 80% da vazão de estado estacionário, foi detectada pelo modelo ANN FF após dois intervalos de amostragem (0,10 h), a partir do instante 8,15 h.

A Figura 5.70 apresenta o resultado da detecção do emperramento F_f pelo modelo de rede neuronal artificial não supervisionada *winner-take-all* (ANN WTA) quando utilizada 80% da vazão de alimentação do reator no estado estacionário, vazão maior que a utilizada no treinamento (70% da vazão no estado estacionário) do método de detecção de falhas.

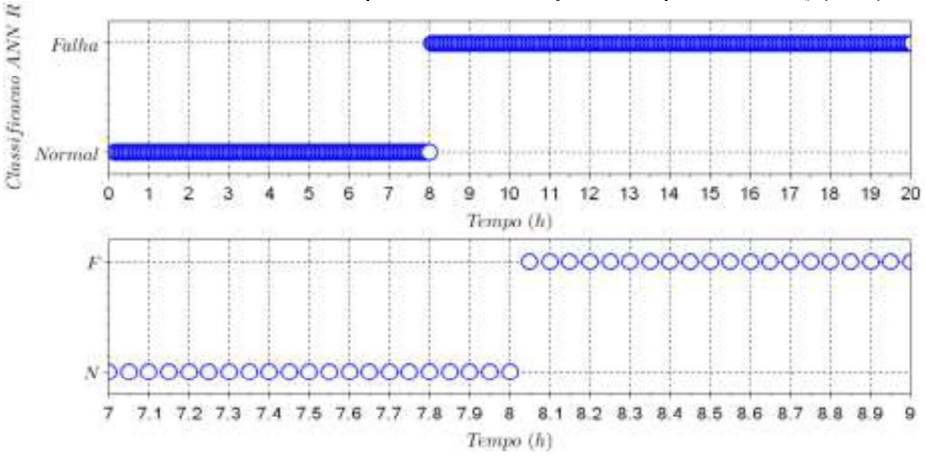
Figura 5.70 - Rótulos obtidos através do processo de detecção do emperramento F_f (80%) - ANN WTA.



O emperramento F_f obtido através da modificação da vazão de alimentação do reator devido a um emperramento da válvula, em que a vazão passa a ser 80% da vazão de estado estacionário, foi detectada pelo modelo ANN WTA após um intervalo de amostragem (0,05 h), mais lentamente um intervalo de amostragem (0,05 h) que o sistema detectou os dados de emperramento F_f utilizados no treinamento (70% da vazão de alimentação do reator no estado estacionário).

A Figura 5.71 apresenta o resultado da detecção do emperramento F_f pelo modelo de rede neuronal artificial de regressão quando a vazão de alimentação é maior (80% da vazão de alimentação do reator no estado estacionário) que a vazão de alimentação apresentada na Seção 4.9.3 (70% da vazão no estado estacionário).

Figura 5.71 - Rótulos obtidos através do processo de detecção do emperramento F_f (80%) - ANN-R.



O emperramento F_f obtido através de 80% da vazão de alimentação do reator no estado estacionário foi detectado pelo modelo ANN-R dentro da mesma quantidade de intervalos de amostragem que os dados com 70% da vazão em estado estacionário. Para ambos os casos, a detecção ocorreu instantaneamente.

A Tabela 5.10 apresenta os instantes de detecção para as diferentes metodologias utilizando redes neuronais artificiais e diferentes amplitudes do emperramento F_f, assim como os índices obtidos para cada situação.

Tabela 5.10 - Índices - ANN FF, ANN WTA e ANN-R - amplitudes emperramento F_f.

F_f	Metodologia	TAD (h)	DF/F	DF/N	DN/N	DN/F
60%	ANN FF	8,10	0,99583	0,00000	1,00000	0,00417
	ANN WTA	**8,05**	**1,00000**	0,00000	1,00000	0,00000
	ANN-R	**8,05**	**1,00000**	0,00000	1,00000	0,00000
70%	ANN FF	8,10	0,99583	0,00000	1,00000	0,00417
	ANN WTA	**8,05**	**1,00000**	0,00000	1,00000	0,00000
	ANN-R	**8,05**	**1,00000**	0,00000	1,00000	0,00000
80%	ANN FF	8,15	0,99167	0,00000	1,00000	0,00833
	ANN WTA	8,10	0,99583	0,00000	1,00000	0,00417
	ANN-R	**8,05**	**1,00000**	0,00000	1,00000	0,00000

Todos os modelos reconheceram as diferentes amplitudes do emperramento F_f. Entre as metodologias ANN de classificação, a ANN de alimentação direta detectou o emperramento F_f mais rapidamente que a ANN *winner-take-all*. Entre as diferentes metodologias, a ANN-R foi a mais rápida a detectar uma operação anormal, detectando instantaneamente a falha.

5.11. Estudo de Caso 2 – Planta Química

Para ilustrar a aplicação da detecção e identificação de falhas através de redes neuronais artificiais supervisionadas de alimentação direta (ANN FF) e não supervisionadas de mapas auto-organizáveis (ANN SOM), foi utilizado os dados simulados de uma planta química composta por dois reatores e um tanque *flash* apresentados anteriormente na Seção 3.6.

5.11.1. ANN Supervisionadas de Alimentação Direta

Para as redes neuronais artificiais de alimentação direta de classificação (*Artificial Neural Networks Feedforward*, ANN FF, ou ANN-C), do mesmo modo que para o SVC, foi necessário treinar a rede neuronal com as informações de treinamento de todas as variáveis controladas, H_1, T_1, H_2, T_2, H_3 e T_3, e variáveis manipuladas, F_{f1}, Q_1, F_{f2}, Q_2, F_R e Q_3 como entradas do modelo ANN de alimentação direta.

Para a saída do modelo ANN FF, foram estabelecidas probabilidades para que cada dado faça parte de cada uma das classes, sendo essas classes as operações faltosas (-13 à -2 e 2 a 13) e as operações normais nos dois estados estacionários (-1 e 1, estado estacionário 1 e estado estacionário 2, respectivamente). Para a saída do treinamento e validação, foram estabelecidas juntamente às variáveis de treinamento e validação o vetor de probabilidades $P = [P_{-13} \quad P_{-12} \quad \cdots \quad P_{-1} \quad P_1 \quad \cdots \quad P_{12} \quad P_{13}]^T$, para $P_i = 1$ se i = falha, e $P_j = 0$ se $j \neq$ falha. As falhas aplicadas -13 à -2 e 2 à 13 seguem as definições apresentadas na Tabela 5.11.

Tabela 5.11 - Probabilidades e falhas aplicadas no estudo de caso 2.

Índice P	Falha Variável_ee	Índice P	Falha Variável_ee
-13	Q3_1	1	ee2
-12	FR_1	2	H1_2
-11	T3_1	3	T1_2
-10	H3_1	4	Ff1_2
-9	Q2_1	5	Q1_2
-8	Ff2_1	6	H2_2
-7	T2_1	7	T2_2
-6	H2_1	8	Ff2_2
-5	Q1_1	9	Q2_2
-4	Ff1_1	10	H3_2
-3	T1_1	11	T3_2
-2	H1_1	12	FR_2
-1	ee1	13	Q3_2

A rede então foi definida com a entrada $x = [H_1 \quad T_1 \quad H_2 \quad T_2 \quad H_3 \quad T_3 \quad F_{f1} \quad Q_1 \quad F_{f2} \quad Q_2 \quad F_R \quad Q_3]^T$ e saída $y = [P_{-13} \quad P_{-12} \quad \cdots \quad P_{-1} \quad P_1 \quad \cdots \quad P_{12} \quad P_{13}]^T$. A estrutura da rede foi definida através de estudo empírico (única camada oculta com o dobro de neurônios da camada de saída) com 12

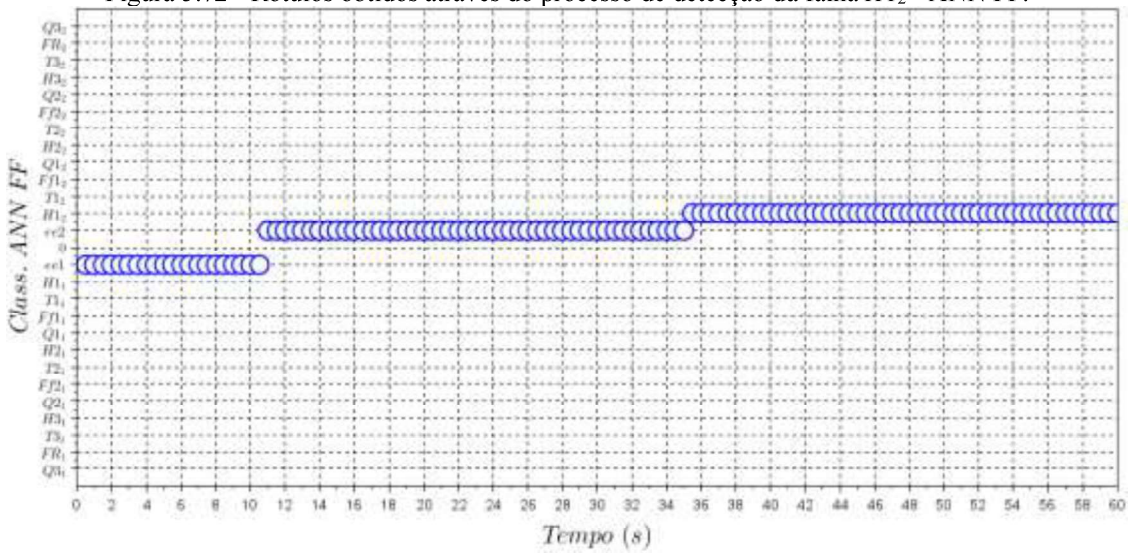
nós na camada de entrada, cada nó para cada uma das variáveis do vetor *x*, com uma camada interna (oculta) de neurônios com 10 nós, número de nós encontrado devido tentativa e erro, e a camada de saída com 26 nós, cada um deles especificando a probabilidade de determinado dado fazer parte de cada uma das classes.

Para o treinamento do sistema de detecção e diagnóstico através de ANN de alimentação direta, foram utilizados dois terços dos dados, enquanto o um terço restante foi utilizado para validação, de acordo com o Capítulo 3. Para cada dois dados selecionados para treinamento, foi selecionado um para validação. Para o treinamento da rede, foram estabelecidas três mil (3000) épocas. Realizando a análise do MSE dos dados de validação do modelo, estabeleceu-se um critério de erro menor que 10^{-5} para definir a quantidade de épocas de treinamento. Cada época no processo de treinamento da rede neuronal é utilizada para melhorar a matriz de pesos **W**, de acordo com a Equação (5.23). Após diversos testes para a seleção da melhor taxa de aprendizagem, para o determinado estudo de caso a taxa de aprendizagem foi estabelecida como 0,9.

Após o treinamento e validação da rede, foi armazenada a matriz peso **W** e as informações pertinentes à rede ANN de alimentação direta, como a estrutura das camadas da rede e a taxa de aprendizado cuja rede foi treinada, que posteriormente foi utilizada para a classificação de novos dados. Para a obtenção da classe dos dados testados foi estabelecida a maior probabilidade de fazer parte de determinada classe para cada instante, e esses dados foram plotados para melhor visualização.

A Figura 5.72 apresenta a classificação dos dados da falha do sensor de nível do tanque 1 (H_1), aplicada no instante de 35 minutos, após a modificação do estado estacionário 1 para o estado estacionário 2 no instante 10 minutos.

Figura 5.72 - Rótulos obtidos através do processo de detecção da falha $H1_2$ - ANN FF.



Para todas as outras falhas, o procedimento foi o mesmo realizado para a falha no sensor de nível do tanque 1. Os resultados obtidos pela detecção de falhas utilizando o modelo ANN FF está apresentado no Apêndice A, da Figura A.53 até a Figura A.78.

5.11.2. ANN Não Supervisionadas de Mapas Auto-Organizáveis

Para as redes neuronais artificiais não supervisionadas de classificação de mapas auto-organizáveis (*Artificial Neural Networks Self-Organizing Maps*, ANN SOM), o treinamento da rede neuronal é realizado com as informações de treinamento de todas as variáveis controladas, H_1, T_1, H_2, T_2, H_3 e T_3, e variáveis manipuladas, F_{f1}, Q_1, F_{f2}, Q_2, F_R e Q_3 como entradas do modelo ANN de alimentação direta.

O modelo ANN SOM não possui variável de saída, destacando por si próprio as características que são importantes na classificação dos dados. Portanto, o único parâmetro a ser fornecido ao método é a estrutura de saída do mapa auto-organizável, sendo uma matriz com a dimensão do número total de classes importantes à classificação

A rede então foi definida com a entrada $x = [H_1 \quad T_1 \quad H_2 \quad T_2 \quad H_3 \quad T_3 \quad F_{f1} \quad Q_1 \quad F_{f2} \quad Q_2 \quad F_R \quad Q_3]^T$ e parâmetro N = [1 26]. A estrutura da rede foi definida através de estudo empírico, sendo o número de falhas e estados estacionários a serem identificados.

Para o treinamento do sistema de detecção e diagnóstico através de ANN SOM, foram utilizados dois terços dos dados, enquanto o um terço restante foi utilizado para validação, de acordo com o Capítulo 3. Para cada dois dados selecionados para treinamento, foi selecionado um para validação. Para o treinamento da rede, foram estabelecidas três mil (3000) épocas.

Após o treinamento e validação da rede, foi armazenada a matriz peso **W** e as informações pertinentes à rede ANN SOM, como estrutura da saída, que posteriormente foi utilizada para a classificação de novos dados. Para a obtenção da classe dos dados testados foi estabelecida a classe para cada instante através da matriz peso **W**, e esses dados foram plotados para melhor visualização.

A elenca os índices apresentados na Seção 3.3 para os diferentes métodos de detecção de falhas utilizando redes neuronais artificiais, com destaque para as ANN supervisionadas (ANN *Feedforward*), que apresentaram bons resultados, apesar do alto custo computacional envolvido no treinamento do método, de diversas horas.

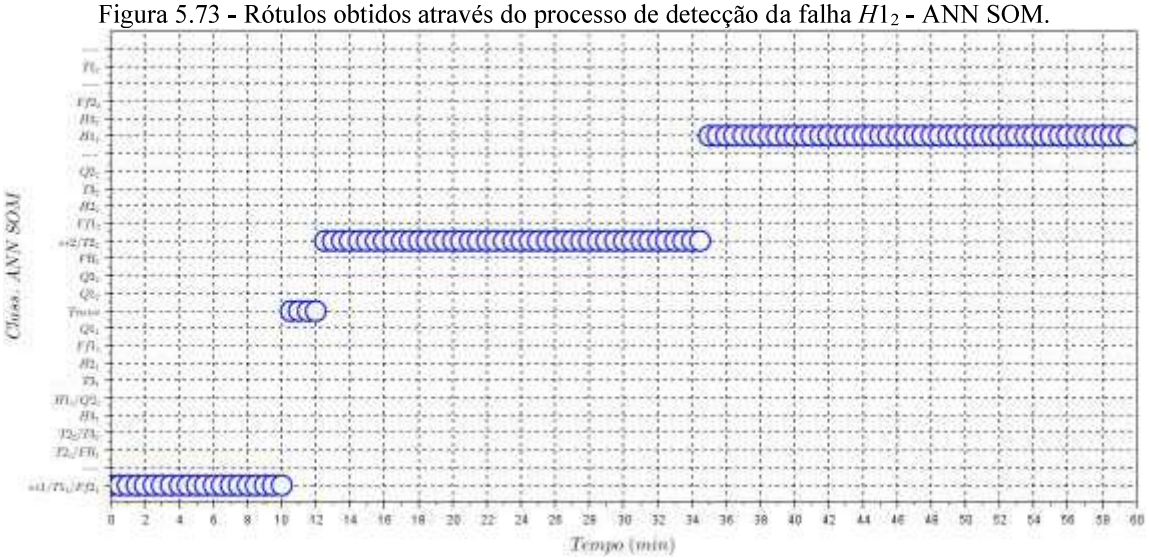
Tabela 5.12 - Índices - Estudo de Caso 2 - ANN FF e ANN SOM.

Oper.	Índices	ANN FF	ANN SOM	Oper.	Índices	ANN FF	ANN SOM
ee1	DF/F	0,9800	0,8000	*ee2*	DF/F	0,9800	0,9200
	DF/N	0,0200	0,2000		DF/N	0,0200	0,0800
	DN/N	0,9857	0,9428		DN/N	0,9857	0,8571
	DN/F	0,0143	0,0572		DN/F	0,0143	0,1429
$H1_1$	DF/F	1,0000	1,0000	$Ff1_1$	DF/F	1,0000	1,0000
	DF/N	0,0000	0,0000		DF/N	0,0000	0,0000
	DN/N	1,0000	1,0000		DN/N	1,0000	1,0000
	DN/F	0,0000	0,0000		DN/F	0,0000	0,0000
$H1_2$	DF/F	1,0000	1,0000	$Ff1_2$	DF/F	1,0000	1,0000
	DF/N	0,0000	0,0000		DF/N	0,0000	0,0000
	DN/N	1,0000	1,0000		DN/N	1,0000	1,0000
	DN/F	0,0000	0,0000		DN/F	0,0000	0,0000
$H2_1$	DF/F	1,0000	1,0000	$Ff2_1$	DF/F	1,0000	*
	DF/N	0,0000	0,0000		DF/N	0,0000	*
	DN/N	1,0000	1,0000		DN/N	1,0000	1,0000
	DN/F	0,0000	0,0000		DN/F	0,0000	0,0000
$H2_2$	DF/F	1,0000	1,0000	$Ff2_2$	DF/F	1,0000	1,0000
	DF/N	0,0000	0,0000		DF/N	0,0000	0,0000
	DN/N	1,0000	1,0000		DN/N	1,0000	1,0000
	DN/F	0,0000	0,0000		DN/F	0,0000	0,0000
$H3_1$	DF/F	1,0000	1,0000	F_{R1}	DF/F	1,0000	1,0000
	DF/N	0,0000	0,0000		DF/N	0,0000	0,0000
	DN/N	1,0000	1,0000		DN/N	1,0000	1,0000
	DN/F	0,0000	0,0000		DN/F	0,0000	0,0000
$H3_2$	DF/F	1,0000	1,0000	F_{R2}	DF/F	1,0000	1,0000
	DF/N	0,0000	0,0000		DF/N	0,0000	0,0000
	DN/N	1,0000	1,0000		DN/N	1,0000	1,0000
	DN/F	0,0000	0,0000		DN/F	0,0000	0,0000
$T1_1$	DF/F	1,0000	*	$Q1_1$	DF/F	1,0000	1,0000
	DF/N	0,0000	*		DF/N	0,0000	0,0000
	DN/N	1,0000	1,0000		DN/N	1,0000	1,0000
	DN/F	0,0000	0,0000		DN/F	0,0000	0,0000
$T1_2$	DF/F	1,0000	0,7800	$Q1_2$	DF/F	1,0000	1,0000
	DF/N**	0,0000	0,2200		DF/N	0,0000	0,0000
	DN/N	1,0000	1,0000		DN/N	1,0000	1,0000
	DN/F	0,0000	0,0000		DN/F	0,0000	0,0000
$T2_1$	DF/F	1,0000	1,0000	$Q2_1$	DF/F	1,0000	1,0000
	DF/N	0,0000	0,0000		DF/N	0,0000	0,0000
	DN/N	1,0000	1,0000		DN/N	1,0000	1,0000
	DN/F	0,0000	0,0000		DN/F	0,0000	0,0000
$T2_2$	DF/F	1,0000	***	$Q2_2$	DF/F	1,0000	1,0000
	DF/N	0,0000	***		DF/N	0,0000	0,0000
	DN/N	1,0000	1,0000		DN/N	1,0000	1,0000
	DN/F	0,0000	0,0000		DN/F	0,0000	0,0000
$T3_1$	DF/F	1,0000	1,0000	$Q3_1$	DF/F	1,0000	*
	DF/N	0,0000	0,0000		DF/N	0,0000	*
	DN/N	1,0000	1,0000		DN/N	1,0000	1,0000
	DN/F	0,0000	0,0000		DN/F	0,0000	0,0000
$T3_2$	DF/F	1,0000	1,0000	$Q3_2$	DF/F	1,0000	1,0000
	DF/N	0,0000	0,0000		DF/N	0,0000	0,0000
	DN/N	1,0000	1,0000		DN/N	1,0000	1,0000
	DN/F	0,0000	0,0000		DN/F	0,0000	0,0000

*O ANN SOM não foi capaz de diferenciar a operação *ee*1 da falha $T1_1$, da falha $Ff2_1$, e da falha $Q3_1$.
** Situação no ANN SOM em que ocorre detecção diferente da falha, porém não é operação normal.
***O ANN SOM não foi capaz de diferenciar a operação *ee*2 da falha $T2_2$.

A Figura 5.73 apresenta a classificação dos dados da falha do sensor de nível do tanque 1 (H_1), aplicada no instante de 35 minutos, após a modificação do estado estacionário 1 para o estado estacionário 2 no instante 10 minutos.

Figura 5.73 - Rótulos obtidos através do processo de detecção da falha $H1_2$ - ANN SOM.



Para todas as outras falhas, o procedimento foi o mesmo realizado para a falha no sensor de nível do tanque 1. Os resultados obtidos pela detecção de falhas utilizando o modelo ANN SOM está apresentado no Apêndice A da Figura A.79 até a Figura A.104.

5.12. Comentários

Analisando a aplicação de diferentes amplitudes de falhas, pode-se afirmar que as redes neuronais artificiais, supervisionadas e não supervisionadas de classificação e de regressão, são capazes de detectar falhas mesmo que não tenham sido treinadas com a mesma amplitude da falha amostrada. Nos casos em que a amplitude da falha a ser detectada é maior que a amplitude dos dados de treinamento, as metodologias são capazes de detectar mais rapidamente as diferentes falhas.

No próximo capítulo, é apresentada a teoria de lógica *fuzzy*, suas aplicações em detecção e diagnóstico de falhas, e os resultados da detecção de dados para os estudos de casos.

CAPÍTULO 6
DETECÇÃO DE FALHAS ATRAVÉS DE LÓGICA DIFUSA

A Lógica Difusa, nebulosa ou *Fuzzy* (ZADEH, 1965) proporciona uma maneira de descrever a imprecisão e a falta de informação agregadas à representação das características de um determinado elemento de um problema específico. A lógica *fuzzy*, conhecida também como lógica difusa, utiliza termos linguísticos, que por definição são imprecisos, de forma análoga à linguagem natural, para apresentar as características de um elemento do problema. A capacidade da lógica difusa compreende o uso de variáveis linguísticas no lugar de variáveis quantitativas, na representação de conceitos imprecisos.

Incorporar a lógica nebulosa em modelos computacionais é útil no caso de sistemas complexos, em que a compreensão do processo é praticamente limitada ao julgamento pessoal, e no caso de processos em que a decisão, a percepção e o raciocínio humano estão envolvidos.

A lógica difusa se torna bastante adequada para o diagnóstico de falhas devido as características apresentadas e é utilizada principalmente na geração de um modelo do sistema ou então na avaliação de resíduos.

Apesar dos dados envolvidos na detecção de dados serem medidas experimentais que possuem incertezas de medida e também ruídos, a lógica difusa foi utilizada como sistema de detecção e identificação de falhas. A lógica difusa não requer um modelo matemático quantitativo, e sim um modelo qualitativo. Dessa forma, o modelo *fuzzy* é livre de erros de modelagem, incertezas e perturbação, fora incertezas e erros inerentes aos dados experimentais. No caso de sistemas complexos, o *fuzzy* tem um papel importante na modelagem porque faz uso do conhecimento do operador do sistema para lidar com o diagnóstico de falhas de maneira mais eficiente.

O sistema decisório do *fuzzy* é descrito baseado nas regras *fuzzy* para estimar a localização da falha (MENDONCA *et al*., 2005). Os subsistemas decisórios do *fuzzy* indicariam a possível causa e decidiriam quando uma falha conhecida ocorre ou não (CHAFI *et al*., 2001). Uma vez que a decisão tomada pela lógica difusa depende do conhecimento humano, falta-lhe uma capacidade de aprendizagem eficaz. A sintonia automática das regras *fuzzy* e das funções de pertinência é difícil, já que o *fuzzy* toma decisões baseadas em dados imprecisos ou ambíguos. De forma a detectar e isolar a falha utilizando a lógica difusa, primeiro o resíduo deve passar pelo processo de "fuzzyficação", avaliado através do mecanismo de inferência *fuzzy*, como as leis SE-ENTÃO, e finalmente os resíduos são "desfuzzyficados". A "fuzzyficação" é o processo que converte os dados das variáveis em escala original para o

conjunto *fuzzy* que contém informações sobre a falha, o mecanismo de inferência *fuzzy* que fornece a relação entre o resíduo e a falha através das regras SE-ENTÃO, e finalmente a "desfuzzyficação" converte a informação de falha *fuzzy* em variáveis em escala original. A decisão final sobre a ocorrência de falhas é então decidida através de inteligência humana.

O principal método de detecção de falhas através da lógica difusa tem sido na avaliação de resíduos (DALTON, 1999). A avaliação de resíduos necessita normalmente de algum tipo de inferência, em que a conclusão de que uma falha ocorre no sistema pode não se basear somente na informação dos resíduos, mas também em informações como registros de manutenção ou experiência do operador, que são dados difíceis de codificar em modelos matemáticos.

Entre algumas das abordagens mais utilizadas citam-se: a clusterização *fuzzy* (*Fuzzy Clustering*), o uso de limiares adaptativos *fuzzy* e o emprego de regras *fuzzy*.

A ideia presente na clusterização *fuzzy* é o reconhecimento de padrões. Dados são utilizados *off line* para determinar os centros dos *clusters* que representam todas as falhas de interesse. Durante a operação do sistema, determina-se o grau de pertinência dos dados coletados a cada um dos *clusters* pré-definidos, sendo o dado classificado naquele *cluster* ao qual ele tem maior pertinência (BEZERRA *et al*., 2013).

O método de clusterização *fuzzy* é útil quando existem muitos resíduos, ou quando não há conhecimento sobre o sistema a ser diagnosticado. Dentre os meios de clusterização *fuzzy* mais utilizados pode-se citar o *Fuzzy* C-*means* e o *Subtractive* (BAKRI; BOUMHIDI, 2018; CHIBANI *et al*., 2018; WU; DONG, 2018; PAN; LI; ZHOU, 2014; GUO; YANG, 2017; BALLESTEROS-MONCADA *et al*., 2015; CHIU, 1996).

Os limiares adaptativos *fuzzy* são capazes de superar as desvantagens de utilizar limiares fixos em variáveis medidas ou em resíduos. Um sistema *fuzzy* pode gerar limiares adaptativos de acordo com as condições de operação da planta observada. Esse método possui a desvantagem de somente indicar a existência de falhas, sendo necessário uma outra etapa de processamento para isolar esta falha.

A maior vantagem da utilização de regras *fuzzy* é possibilitar a inclusão de informação heurística no esquema de análise. Regras *fuzzy* tendem a facilitar o entendimento e funcionam de modo semelhante ao utilizado por seres humanos para resolver problemas.

A análise de resíduos empregando regras *fuzzy* faz com que cada resíduo pode ser zero, negativo ou positivo com um certo grau de pertinência, e que um sistema de inferência pode ser usado para determinar o grau de presença de uma determinada falha no sistema ou para determinar o grau de gravidade de uma falha presente no sistema. As regras *fuzzy* indicam as

condições nas quais falhas existem e também indicam as condições nas quais não há falhas. Por exemplo:

- SE resíduo1 é POSITIVO E resíduo2 é NEGATIVO
  ENTÃO falha1 é PRESENTE
- SE resíduo1 é ZERO E resíduo2 é ZERO
  ENTÃO falha1 é AUSENTE

Pode-se observar que neste tipo de sistema o efeito da falha nos resíduos deve ser conhecido.

Também é possível determinar a base de regras de forma automática a partir de dados do sistema em consideração, mas esta abordagem possui algumas desvantagens: dados referentes a todas as falhas de interesse, e em todos os pontos de operação, devem estar disponíveis; algoritmos de busca de regras podem ficar presos em mínimos locais e a base encontrada pode não representar corretamente o sistema, e, se estas regras não refletirem a experiência do operador, estas podem não ser validadas.

A alternativa que pode ser utilizada é iniciar o processo de criação de regras através de uma base inicial elaborada pelo operador.

A sequência de passos a seguir apresenta o procedimento:

1. Determine o comportamento dos resíduos em relação a cada falha;
2. Extraia as regras que expressem esse comportamento e adicione-as à base existente;
3. Verifique se todas as falhas podem ser isoladas. Caso afirmativo, prossiga ao passo 4. Se não:
   a. Adicione mais funções de pertinência ou
   b. Adicione outro resíduo ao sistema e volte para 1;
4. Uma vez que a base de regras está completa, os parâmetros da função de pertinência podem ser encontrados;
5. A base de regras resultante pode ser testada. Se o resultado for satisfatório, o processo está completo. Se não, retorne ao passo 3.

Existem diversos métodos que podem ser usados para sintonizar os parâmetros do sistema *fuzzy*, incluindo Clusterização *Fuzzy*, Redes Neuronais e Algoritmos Genéticos (AMARAL *et al.*, 2003; ABDULGHAFOUR; EL-GAMAL, 1996; NAUCK; KRUSE, 1995; GUNN *et al*., 1997);

6.1. *Fuzzy* C-means

Para realizar a detecção de falhas através de classificação utilizando a lógica difusa, um dos métodos que podem ser utilizados é o *Fuzzy clustering*. O método de *fuzzy clustering* envolve o estabelecimento de *clusters* ou centros, que podem ser identificados através das diferentes classes de dados classificados. Esses centros são escolhidos de forma que seja maximizado a distância entre estes, de forma que os dados, quando classificados, podem fazer parte de mais um centro.

O *Fuzzy C-means* (FCM) (DUNN, 1973; BEZDEK, 1981), um dos algoritmos de "clusterização" mais amplamente utilizado, consiste no seguinte procedimento:

1. Seleciona-se o número de *clusters*, no nosso caso, o número de operações (normal e falhas);
2. Atribui-se coeficientes aleatoriamente para cada ponto que faz parte do *cluster*;
3. Repete-se até que o algoritmo converge (ou seja, a modificação de coeficientes entre duas iterações não deve ser maior que ε, o limite de sensibilidade estabelecido;
4. Calcula-se o centroide para cada *cluster*;
5. Para cada ponto, calcula-se o coeficiente de pertinência para pertencer em cada *cluster*;

Para o cálculo do centroide, para qualquer x existirá um conjunto de coeficientes que fornecem o grau de pertinência para o k-ésimo *cluster*, $w_k(\mathbf{x})$. Para o *fuzzy c-means*, o centroide de um cluster é a média de todos os pontos, ponderados pelo grau de pertinência de determinado *cluster*, ou de forma matemática (Equação (6.1)).

$$\mathbf{c}_k = \frac{\sum_x w_k(\mathbf{x})^m \mathbf{x}}{\sum_x w_k(\mathbf{x})^m} \tag{6.1}$$

em que m é o hiper-parâmetro que controla o quão nebuloso o *cluster* será, ou o quanto espalhado os dados estarão entre os *clusters*. Quanto maior m, mais nebuloso será o *cluster*.

O algoritmo FCM realiza a partição de um conjunto finito de n elementos $X = \{\mathbf{x}_1, \dots, \mathbf{x}_n\}$ em uma coleção de c *fuzzy clusters* que respeitam os critérios de sensibilidade e espalhamento. Fornecido um conjunto de dados finitos, o algoritmo retorna uma lista de c centros de *clusters* $C = \{\mathbf{c}_1, \dots, \boldsymbol{c}_c\}$ e uma matriz de partição $W = w_{i,j} \in [0,1], i = 1, \dots, n, j = 1, \dots, c$, em que cada elemento, $w_{i,j}$, apresenta o grau que cada elemento, $\mathbf{x}_i$, pertence ao *cluster* $\boldsymbol{c}_j$. O algoritmo FCM procura minimizar a seguinte função objetivo, apresentada pela Equação (6.2).

$$\arg\min_{C} \sum_{i=1}^{n} \sum_{j=1}^{c} w_{i,j}^{m} \left\| \mathbf{x}_i - \mathbf{c}_j \right\|^2 \tag{6.2}$$

em que:

$$w_{i,j} = \frac{1}{\sum_{k=1}^{c} \left(\frac{\left\| \mathbf{x}_i - \mathbf{c}_j \right\|}{\left\| \mathbf{x}_i - \mathbf{c}_k \right\|} \right)^{\frac{2}{m-1}}} \tag{6.3}$$

A lógica difusa pode ser utilizada juntamente a outras técnicas de detecção de falhas, entre elas as máquinas de vetores de suporte e as redes neuronais artificiais.

6.2. Lógica *Fuzzy*

A detecção de falhas através da lógica difusa faz uso de funções de pertinência para as variáveis de entrada do sistema de detecção, cujos valores são fuzzyficados. Após a fuzzyficação, esses dados passam pelo procedimento de inferência, com informações da base de regras de classificação, como apresentado na Figura 6.1. Após a inferência, os dados são desfuzzyficados através da função de pertinência da variável de saída do sistema de detecção.

Figura 6.1 - Estrutura do classificador *fuzzy*.

Um conjunto *fuzzy A* definido no universo de discurso *X* é caracterizado por uma função de pertinência μ_A, a qual mapeia os elementos de *X* para o intervalo [0,1]. Dessa forma, a função de pertinência associa a cada elemento *x* pertencente a *X* um número real $\mu_A(x)$ no intervalo [0,1], que representa o grau de pertinência do elemento *x* ao conjunto *A*, isto é, o quanto é possível para o elemento *x* pertencer ao conjunto *A*.

Um conjunto *fuzzy* A em X é expresso como um conjunto de pares ordenados (Equação (6.4)).

$$A = \{(x, \mu_A(x)) | x \in X\} \tag{6.4}$$

6.2.1. Fuzzyficação

A etapa de fuzzyficação é relacionada a definição de atributos, ou seja, a definição das variáveis linguísticas, que por sua vez estão relacionadas ao estado do processo, caracterizadas pelas variáveis de entrada do classificador *fuzzy*. Cada variável linguística é associada a um universo de discurso, sendo que a partição deste universo é feita de acordo com os valores que a variável pode assumir. Estes valores correspondem aos conjuntos *fuzzy*.

Através do estágio de fuzzyficação os valores observados, normalizados ou não, das variáveis de entrada são associados ao respectivo universo de discurso, possibilitando a obtenção do grau de pertinência aos conjuntos *fuzzy* associados a cada variável (ZADEH, 1973). Por exemplo, considere $x_0 \in R$ fuzzificado, nos conjuntos A_i, o resultado do estágio de fuzzificação é um vetor no intervalo $[0, 1]^n$. Este vetor apresenta os valores dos graus de pertinência para cada conjunto A_i, da seguinte forma: $[\mu_{A_1}(x_0), \mu_{A_2}(x_0), \mu_{A_3}(x_0), \ldots, \mu_{A_n}(x_0)]$, em que A_1, A_2, A_3, ..., A_n correspondem a conjuntos *fuzzy* da interface de entrada. Estes conjuntos são conhecidos como partição do universo de discurso (PEDRYCZ, 1994).

6.2.2. Base de regras

A base de regras é formada por um conjunto de instruções *fuzzy* que, quando processadas, produzem uma solução aproximada para um determinado problema. O conjunto de regras *fuzzy* deve refletir a meta do classificador (ZADEH, 1973). A definição do conjunto de regras *fuzzy*, geralmente, baseia-se na experiência dos operadores e conhecimento do engenheiro de processo. Contudo, de acordo com Santos e colaboradores (2004), encontram-se na literatura estudos que buscam a definição das regras *fuzzy* para diferentes aplicações a partir do processo de aprendizado.

6.2.3. Estágio de inferência

O classificador *fuzzy* utiliza o contexto de inferência *fuzzy* e de acordo com a regra de implicação proposicional *modus ponens*, que pode ser definida como o processo que fará com que os dados sejam classificados através da base de regras. Por exemplo, se na entrada de um sistema X é A' e Y é B', e a base de regras diz que: se X é A_i e Y é B_i então Z é C_i, a principal consequência da entrada do sistema é que Z é C', em que X e Y são variáveis linguísticas que relatam o estado do processo, Z é a variável linguística de classificação, A', A_i, B', B_i, C', C_i são conjuntos *fuzzy* de X, Y e Z nos universos de discurso U, V e W, respectivamente.

A regra *modus ponens* fornecerá a consequência através dos dados de entrada e da premissa que é definida pela parte condicional da regra (JACQUES *et al*., 2002). O conjunto de diferentes regras *fuzzy* produz o comportamento do classificador nebuloso. Para a implementação de cada regra e obtenção da função resultante, utilizam-se operadores lógicos, cuja seleção deve ser levada a partir de uma avaliação de qual o melhor operador para cada caso. Para implementar a função de implicação *fuzzy* são empregadas, geralmente, a regra de operação definida por Mamdani que utiliza o operador interseção e a regra de operação definida por Takagi-Sugeno.

6.2.4. Desfuzzyficação

O algoritmo de classificação faz com que o processamento das variáveis linguísticas de entrada resulte em um valor da variável linguística de saída. Assim, o processo de desfuzzyficação consiste em selecionar um valor numérico específico que represente o resultado *fuzzy* da variável de saída produzido pelo conjunto de regras *fuzzy*. A escolha de um método de desfuzzyficação pode ter um impacto significativo na resposta produzida por um controlador *fuzzy*.

Os métodos de desfuzzyficação que são utilizados frequentemente são: Critério Máximo, Método do Centro de Gravidade, Bissetor da Área, Métodos dos Máximos e Mínimo do Máximo (JACQUES *et al*., 2002).

6.3. Funções de Pertinência

Comparando a lógica clássica (booleana) com a lógica difusa, na lógica clássica os valores lógicos das variáveis podem ser apenas 0, correspondente ao valor falso, e 1, correspondente ao valor verdadeiro, enquanto que a lógica difusa foi estendida para lidar com o conceito de verdade parcial, o calor verdade pode compreender entre completamente verdadeiro e completamente falso. Portanto, enquanto um conjunto ordinário tem dois valores para a função característica, 0 ou 1, os conjuntos difusos têm o intervalo $0 \leq x \leq 1$ (ZADEH, 1965). A função que associa cada elemento de um conjunto difuso A a medida de sua pertinência a esse conjunto é definida como função de pertinência. Nessa seção são apresentados os principais tipos de funções de pertinência que podem ser utilizadas para caracterizar os conjuntos difusos, juntamente com características dessas funções.

A função de pertinência, $\mu(x)$, é a responsável por associar os números reais contidos no intervalo $0 \leq x \leq 1$ aos elementos $x \in A$, resultando assim o grau de pertinência do elemento x no conjunto A. A pertinência de um elemento em relação a determinado conjunto deve ser entendida como a gradação com que este elemento está relacionado ao conjunto.

Definir função de pertinência é subjetiva, pois uma função definida para um mesmo conceito pode apresentar resultados diversos se definida por diferentes pessoas. Além de definir o quanto certo elemento pertence a determinado conjunto, a função também serve para representar os limites de um conjunto difuso. Para determinar a maneira que varia e os limites de cada um destes conjuntos, é necessário que a função de pertinência seja adequada.

As funções de pertinência dos conjuntos difusos podem ter diversas formas, sendo a função triangular, um dos tipos mais utilizados. Usualmente, uma função trapezoidal é expressa através de quatro parâmetros, que descrevem os quatro vértices do trapézio que representa a função. Na função trapezoidal existem dois parâmetros com pertinência igual a 1, ao passo que na função triangular existe somente um parâmetro com esse grau de pertinência (SANTOS *et al*., 2004). A função de pertinência triangular é frequentemente utilizada em diversas aplicações dos conjuntos difusos: modelos *fuzzy*, controladores e classificação de padrões. Uma justificativa óbvia para a utilização desta função seja a simplicidade da mesma.

6.3.1. Tipos de funções de pertinência

Segundo Gupta e colaboradores (2003), existem alguns tipos principais de funções de pertinência:

- Função triangular;
- Função trapezoidal;
- Função gaussiana;
- Função sino;
- Função sigmoidal.

Outras funções de pertinência podem ser construídas através das funções de pertinência principais.

Nas seções seguintes cada uma dessas funções é detalhada, juntamente com a indicação dos parâmetros que as definem. Com relação às expressões representativas das funções, X corresponde ao universo de discurso de cada variável linguística, sendo x os valores pertencentes a X.

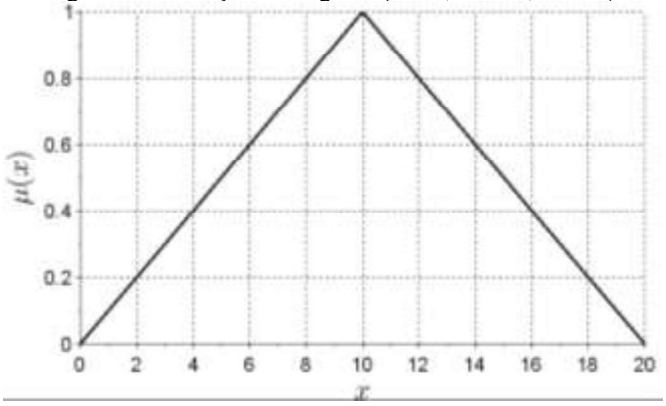
Função de Pertinência Triangular

A curva triangular é uma função de x e depende de três parâmetros escalares $[a, b, c]$. A função é caracterizada pela Equação (6.5):

$$f(x: a, b, c) = \begin{cases} 0 & , \ se \quad x \leq a \quad ou \quad x \geq c \\ \dfrac{x-a}{b-a} & , \ se \quad x > a \quad e \quad x < b \\ 1 & , \ se \quad x = b \\ \dfrac{c-x}{c-b} & , \ se \quad x > b \quad e \quad x < c \end{cases} \tag{6.5}$$

Os parâmetros a e c correspondem aos valores de X localizados nos vértices da base do triângulo, que possuem pertinência igual a zero. O parâmetro b é o valor de X localizado no vértice com a pertinência igual a 1 (Figura 6.2).

Figura 6.2 - Função Triangular (a = 0; b = 10; c = 20).



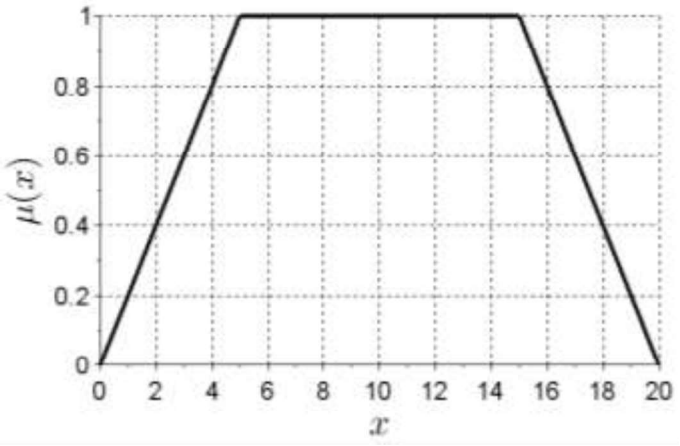
Função de Pertinência Trapezoidal

A curva trapezoidal é uma função de x e depende de quatro parâmetros escalares $[a, b, c, d]$. A função é caracterizada pela Equação (6.6):

$$f(x: a, b, c) = \begin{cases} 0 & , \ se \quad x \leq a \quad ou \quad x \geq d \\ \dfrac{x-b}{b-a} & , \ se \quad x > a \quad e \quad x < b \\ 1 & , \ se \quad b \leq x \leq c \\ \dfrac{d-x}{d-c} & , \ se \quad x > b \quad e \quad x < c \end{cases} \tag{6.6}$$

Os parâmetros a e d correspondem aos valores de X localizados nos vértices da base do triângulo, que possuem pertinência igual a zero. O parâmetro b e c são os valores de X localizados nos vértices com a pertinência igual a 1 (Figura 6.3).

Figura 6.3 - Função Trapezoidal (a = 0; b = 5; c = 15; d = 20).



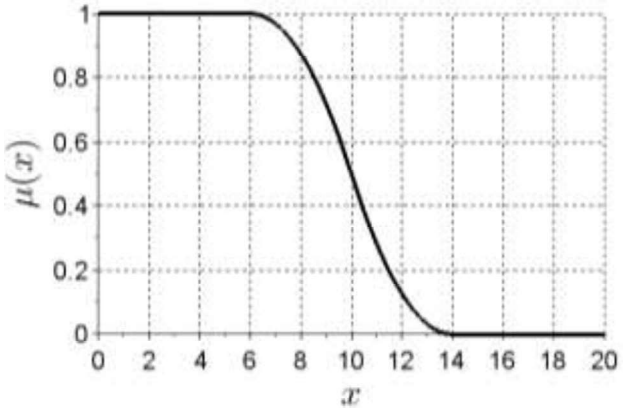
Função de Pertinência Forma de Z

A curva de forma de Z é uma função de x e depende de dois parâmetros escalares $[a, b]$. A função é caracterizada pela Equação (6.7):

$$f(x: a, b) = \begin{cases} 1 \quad , \; se \quad a \geq b \quad e \quad x \leq \dfrac{a+b}{2} \\ 0 \quad , \; se \quad a \geq b \quad e \quad x > \dfrac{a+b}{2} \\ 1 \quad , \; se \quad a < b \quad e \quad x \leq a \\ 1 - 2\left(\dfrac{x-a}{b-a}\right)^2 , \; se \quad a < b \quad e \quad a < x \leq \dfrac{a+b}{2} \\ 2\left(\dfrac{b-x}{b-a}\right)^2 , \; se \quad a < b \quad e \quad \dfrac{a+b}{2} < x \leq b \\ 0 \quad , \; se \quad a < b \quad e \quad x > b \end{cases} \tag{6.7}$$

Os parâmetros a e b correspondem aos valores de X localizados nos extremos da parte inclinada da curva. Para $x \leq a$, $\mu(x) = 1$ e para $x \geq b$, $\mu(x) = 0$ (Figura 6.4).

Figura 6.4 - Função Forma de Z (a = 6; b = 14).



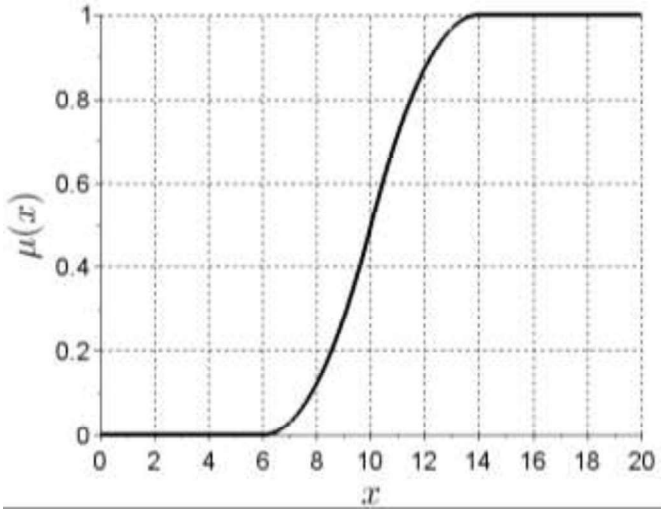
Função de Pertinência Forma de S

A curva de forma de S é uma função de x e depende de dois parâmetros escalares $[a, b]$. A função é caracterizada pela Equação (6.8):

$$f(x: a, b) = \begin{cases} 1 \quad , \; se \quad a \geq b \quad e \quad x \geq \dfrac{a+b}{2} \\ 0 \quad , \; se \quad a \geq b \quad e \quad x < \dfrac{a+b}{2} \\ 0 \quad , \; se \quad a < b \quad e \quad x \leq a \\ 2\left(\dfrac{x-a}{b-a}\right)^2 , \; se \quad a < b \quad e \quad a < x \leq \dfrac{a+b}{2} \\ 1 - 2\left(\dfrac{b-x}{b-a}\right)^2 , \; se \quad a < b \quad e \quad \dfrac{a+b}{2} < x \leq b \\ 1 \quad , \; se \quad a < b \quad e \quad x > b \end{cases} \tag{6.8}$$

O parâmetro a é o maior valor do universo de discurso que possui pertinência 0, e b é o menor valor do universo de discurso que possui pertinência 1, ou seja, para $x \leq a$, $\mu(x) = 0$ e para $x \geq b$, $\mu(x) = 1$ (Figura 6.5).

Figura 6.5 - Função Forma de S (a = 14; b = 6).

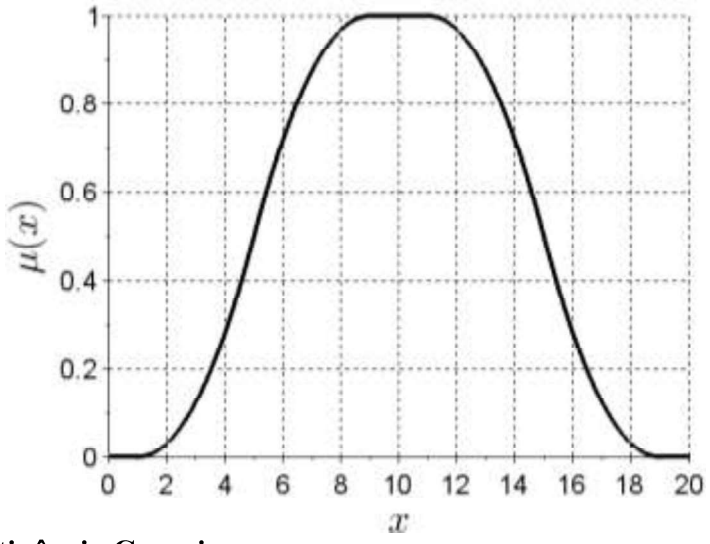


Função de Pertinência Forma de π

A curva de forma de π é uma função de x e depende de quatro parâmetros escalares $[a, b, c, d]$. Os parâmetros a e d estão localizados na base da curva enquanto b e c estão localizados na parte do domínio com pertinência igual a 1.

Zadeh (1973) definiu a expressão da função π como sendo a expressão de duas funções forma de S acopladas, onde os pontos com parâmetros b e c coincidem, ou seja, segundo Zadeh (1973) o gráfico da função não apresenta um intervalo de valores ($b \leq x \leq c$) no universo de discurso com pertinência igual a 1. Contudo, deve-se ressaltar que as funções são as mesmas. A Figura 6.6 apenas apresenta mais de um valor com pertinência igual a 1.

Figura 6.6 - Função Forma de π (a = 1; b = 9; c = 11; d = 19).



Função de Pertinência Gaussiana

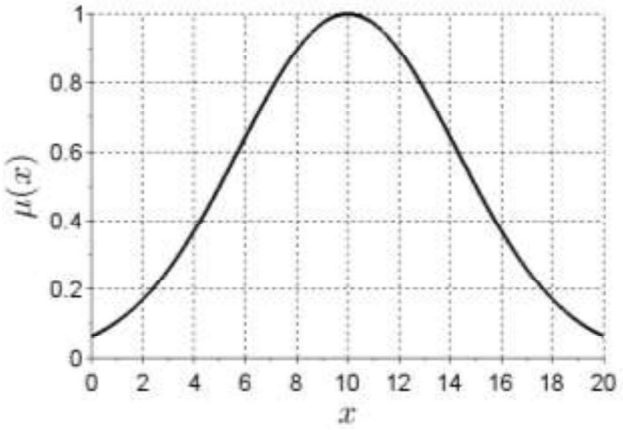
A função simétrica Gaussiana depende de dois parâmetros, σ e c, caracterizada pela Equação (6.9).

$$f(x, \sigma, c) = e^{\frac{-(x-c)^2}{2\sigma^2}} \tag{6.9}$$

em que: σ = desvio padrão e c = média.

A Figura 6.7 mostra a forma de uma função Gaussiana.

Figura 6.7 - Função Gaussiana (σ = 4,247; c = 10).



Os parâmetros da função gaussiana estão listados em ordem no vetor [desvio padrão, média]. Ela é a base para a determinação da expressão da função combinação de duas gaussianas. Para facilitar a utilização das funções gaussiana e combinação de duas gaussianas, foi considerado que a expressão da função é composta por $f(x) = e^{-ax^2}$, uma vez que o parâmetro c está presente, para determinar o quanto a função será deslocada do eixo y para esquerda ou direita. Assim, o parâmetro a será considerado como: $\frac{1}{2\sigma^2}$. Realizando a primeira e

a segunda derivada da expressão $f(x) = e^{-ax^2}$ conclui-se que a função é crescente para valores menores que zero e decrescente para valores maiores que zero. Os pontos $\left\{\left(-\sigma;\frac{1}{\sqrt{e}}\right)\left(\sigma;\frac{1}{\sqrt{e}}\right)\right\}$ são os pontos em que a função muda de concavidade, sendo determinados ao realizar a segunda derivada da função $f(x) = e^{-ax^2}$ e igualando-a a zero.

Conhecendo o comportamento da função, pode-se realizar algumas considerações, a fim de se determinar a relação entre os parâmetros média e desvio padrão. Admitindo-se que sejam conhecidos a média e os valores de x para os quais a função de pertinência é igual a 0,5, as determinações dos parâmetros são: c = média e $\sigma = \sqrt{(A-c)^2/1{,}38}$ em que: A é um valor de x que possui pertinência igual a 0,5.

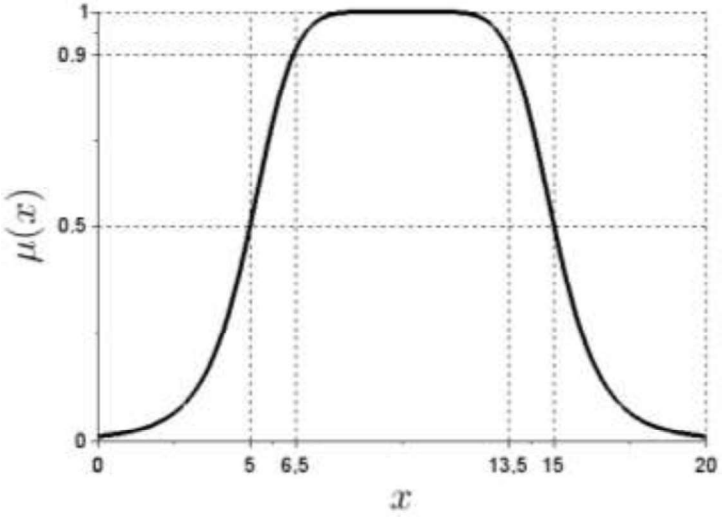
Função de Pertinência Forma de Sino

A função Sino depende de três parâmetros, a, b e c, e é definida pela Equação (6.10).

$$f(x: a, b, c) = \frac{1}{1 + \left|\frac{x-c}{a}\right|^{2b}} \tag{6.10}$$

O parâmetro b é usualmente positivo. O parâmetro c está localizado no centro da curva, que é mostrada na Figura 6.8.

Figura 6.8 - Função Forma de Sino (a = 5; b = 3.278;c = 10).



A definição da curva se dá pelos dois pontos que a curva assume pertinência igual a 0,5 e pelos dois pontos em que a pertinência é igual a 0,9. Chamando x_1 e x_2 os pontos em que a pertinência é 0,5, observa-se através da curva traçada que: $x_1 = c - a$ e $x_2 = c + a$. Dado que c é conhecido (centro da curva): $a = c - x_1$ ou $a = x_2 - c$. A partir daí, conhecendo-se os

valores de x para os quais a pertinência é 0,9, é possível determinar o valor do parâmetro b. Seja x_3 e x_4 os valores de X com pertinência igual a 0,9. Então (Equação (6.11):

$$b = \frac{-1{,}0986}{ln\left|\frac{x_3 - c}{a}\right|^{2b}} \quad ou \quad b = \frac{-1{,}0986}{ln\left|\frac{x_4 - c}{a}\right|^{2b}} \tag{6.11}$$

Considerando a curva apresentada na Figura 6.8, em que: a = 5, c = 10, x_3 = 6,4 e x_4 = 13,6 (valores de x_3 e x_4 aproximados através de interpolação linear), o valor de b é aproximadamente 3,278.

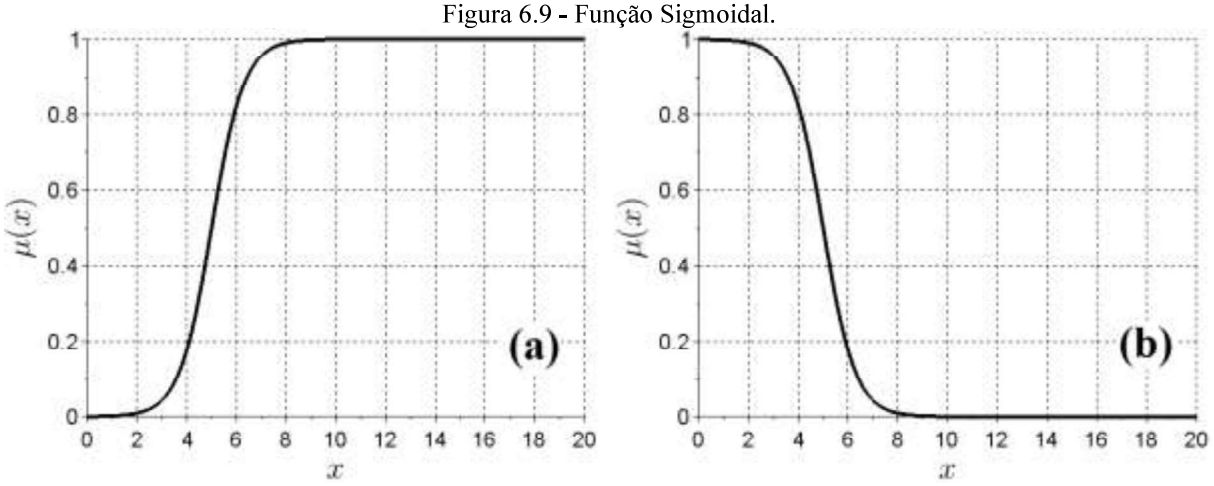
Função de Pertinência Sigmoidal

A função Sigmoidal serve de primitiva para outras duas funções: função diferença entre duas sigmoidais e função produto de duas sigmoidais. A função sigmoidal depende de dois parâmetros, a e b, e é definida pela Equação (6.10).

$$f(x, a, b) = \frac{1}{1 + e^{-a(x-b)}} \tag{6.12}$$

Dependendo do sinal do parâmetro a a função de pertinência sigmoidal é inerentemente aberta à direita ou à esquerda, ou seja, se $a > 0$, $f(x)$ é crescente; se se $a < 0$, $f(x)$ é decrescente. Assim, quando se $x = 0$, a curva muda de concavidade $f(x)$ é crescente. A função de pertinência é apropriada para representar concepções como “muito grande” ou “muito negativo”. Observando a função, constata-se que o parâmetro b é responsável por deslocar a curva para a direita ou esquerda do eixo das ordenadas e é o valor de X para o qual a pertinência é igual a 0,5, enquanto $|a|$ corresponde a velocidade a curva tende para zero. A Figura 6.9 apresenta algumas variações da forma da função sigmoidal, a fim de determinar a relação entre os parâmetros média e desvio padrão.

Figura 6.9 - Função Sigmoidal.

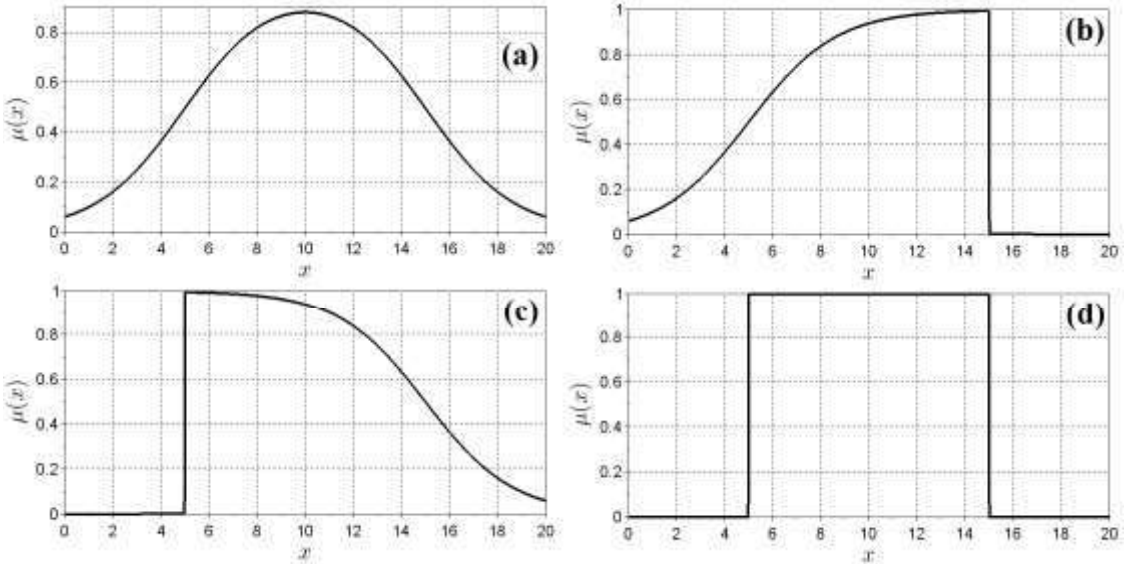


(a) $a > 0$: a = 1,543; b = 5; (b) $a < 0$: a = -1,543; b = 5;

Função de Pertinência Diferença Absoluta entre duas Sigmoidais

A função de pertinência composta pela diferença absoluta entre duas sigmoidais depende de quatro parâmetros, $[a, b, c, d]$, e será determinada pela diferença entre duas funções sigmoidais. Os marcadores externos b e d determinam os valores de $f(x)$ para os quais a pertinência é igual a 0,5. A Figura 6.10 apresenta diferentes formas da função de pertinência em função de valores assumidos para a e c.

Figura 6.10 - Variações do Gráfico da Função Diferença Absoluta entre Duas Sigmoidais.



(a) parâmetros positivos; (b) $a \to \infty$; (c) $c \to \infty$; (d) a e $c \to \infty$.

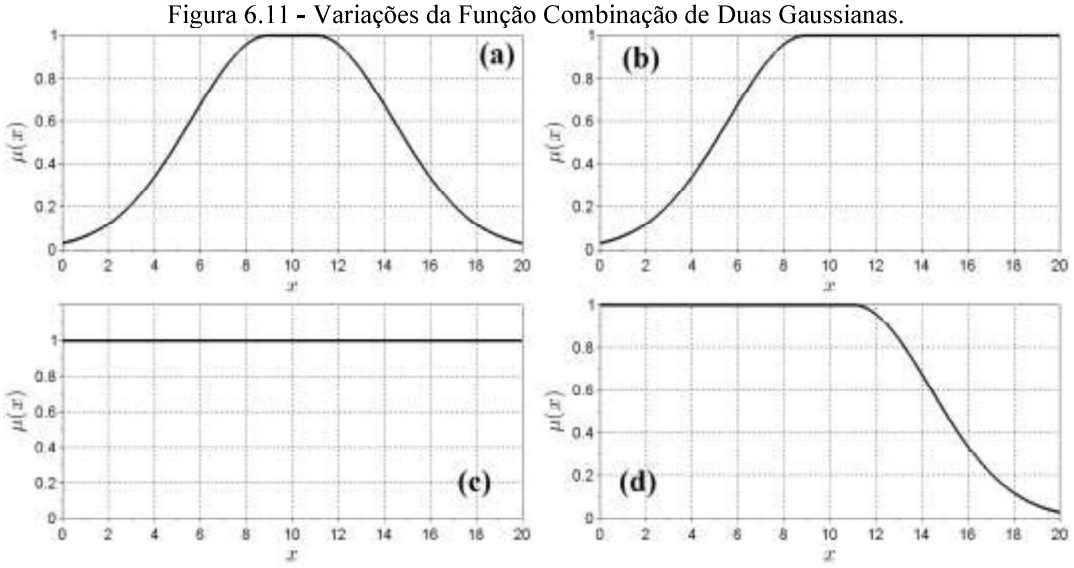
A Equação (6.13) apresenta a função de pertinência diferença absoluta de duas sigmoidais.

$$f(x, a, b, c, d) = \left| \frac{1}{1 + e^{-a(x-b)}} - \frac{1}{1 + e^{-c(x-d)}} \right| \tag{6.13}$$

Função de Pertinência Combinação de duas Funções Gaussianas

A função de pertinência composta pela combinação de duas gaussianas depende de quatro parâmetros, $[\sigma_1, c_1, \sigma_2, c_2]$. A primeira função é especificada pelos parâmetros σ_1 e c_1. Eles determinam a forma da curva mais à esquerda. A segunda função é especificada pelos parâmetros σ_2 e c_2 que determinam a forma da curva mais à direita. As variações da forma dessa curva são mostradas na Figura 6.11.

Figura 6.11 - Variações da Função Combinação de Duas Gaussianas.



(a) parâmetros positivos; (b) $\sigma_1 \to \infty$; (c) σ_1 e $\sigma_2 \to \infty$; (d) $\sigma_2 \to \infty$.

A Equação (6.14) apresenta a função de pertinência combinação de duas gaussianas, ou função de pertinência gaussiana extendida.

$$f(x, \sigma_1, c_1, \sigma_2, c_2) = \begin{cases} e^{\frac{-(x-c_1)^2}{2\sigma_1^2}}, & x \leq c_1 \\ 1, & c_1 \leq x \leq c_2 \\ e^{\frac{-(x-c_2)^2}{2\sigma_2^2}}, & x \geq c_2 \end{cases} \tag{6.14}$$

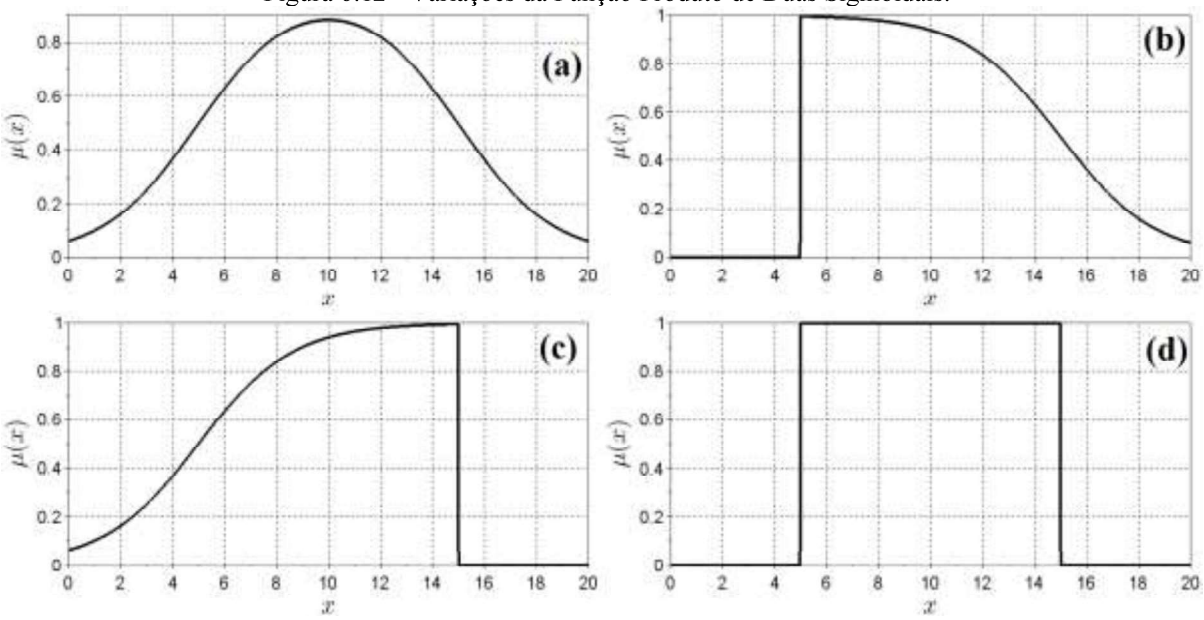
Função de Pertinência Produto de duas Funções Sigmoidais

A função de pertinência produto de duas sigmoidais depende de quatro parâmetros, $[a, b, c, d]$. Ela será determinada pelo produto de duas funções sigmoidais. Os marcadores externos b d determinam os valores de $f(x)$ para os quais a pertinência é igual a 0,5. A Equação (6.15) apresenta a função de pertinência produto de duas sigmoidais.

$$f(x, a, b, c, d) = \frac{1}{1 + e^{-a(x-b)}} \cdot \frac{1}{1 + e^{-c(x-d)}} \tag{6.15}$$

As variações da forma da função de pertinência produto de duas funç são mostradas na Figura 6.12.

Figura 6.12 - Variações da Função Produto de Duas Sigmoidais.



(a) parâmetros positivos; (b) $\sigma_1 \to \infty$; (c) $\sigma_2 \to \infty$; (d) σ_1 e $\sigma_2 \to \infty$.

6.4. Máquinas de Vetores de Suporte *Fuzzy*

A máquina de vetores de suporte *fuzzy* é uma técnica de classificação que passou a ter atenção no meio acadêmico desde a última década (LIN; WANG, 2002; LIU; ZIO, 2018). A teoria de classificação das máquinas de vetores de suporte (*Support Vector Machines* - SVM) *fuzzy* é baseada na ideia da minimização do risco estrutural (*Structural Risk Minimization* - SRM) (VAPNIK, 1995). Um grau de pertinência *fuzzy* é aplicado a cada entrada da SVM, reformulando o sistema SVM em *Fuzzy* SVM (*Fuzzy Support Vector Machines* - FSVM) de forma que diferentes entradas podem realizar diferentes contribuições ao aprendizado da superfície de decisão. O FSVM melhora o SVM na redução de efeitos de *outliers* e ruídos nos dados. O FSVM é aplicável em casos que os dados possuem características impossíveis de serem modeladas.

Considerando um conjunto de dados de treinamento *S* classificados associados ao grau de pertinência *fuzzy* (Equação (6.16)),

$$(y_1, \mathbf{x}_1, s_1), \cdots, (y_i, \mathbf{x}_i, s_i) \tag{6.16}$$

em que para cada ponto de treinamento $\mathbf{x}_i \in \mathcal{R}^N$ é estabelecido um rótulo $y_i \in \{-1,1\}$ e um grau de pertinência *fuzzy* $\sigma \leq s_i \leq 1$, com $i = 1, \ldots, l$, e $\sigma > 0$ suficientemente pequeno. Seja $\mathbf{z} = \varphi(\mathbf{x})$ o mapeamento φ do espaço de vetores correspondente de $\mathcal{R}^N$ para o espaço característico $\mathcal{Z}$.

Uma vez que o grau de pertinência s_i é o comportamento do ponto correspondente $\mathbf{x}_i$ para uma classe e o parâmetro ξ_i é a medida de erro do SVM, o termo $s_i\xi_i$ é a medida de erro com peso diferente. O problema de hiperplano ótimo é então reduzido a solução da Equação (6.17),

$$\begin{gathered} \min \frac{1}{2}\mathbf{w}\cdot\mathbf{w} + C\sum_{i=1}^{l} s_i\xi_i \\ sujeito\ a\ y_i(\mathbf{w}\cdot\mathbf{z}_i + b) \geq 1 - \xi_i, i = 1, \ldots, l \\ \xi_i \geq 0, \qquad i = 1, \ldots, l \end{gathered} \tag{6.17}$$

em que C é uma constante. Fica claro que quanto menor o grau de pertinência s_i, menor será o efeito do parâmetro ξ_i da Equação B de forma que o ponto correspondente $\mathbf{x}_i$ tem menor importância.

Para resolver o problema de otimização, é construído o Lagrangiano (Equação (6.18))

$$\boldsymbol{L}(\mathbf{w}, b, \xi, \alpha, \beta) = \frac{1}{2}\mathbf{w}\cdot\mathbf{w} + C\sum_{i=1}^{l} s_i\xi_i - \sum_{i=1}^{l}\alpha_i(y_i(\mathbf{w}\cdot\mathbf{z}_i + b) - 1 + \xi_i) - \sum_{i=1}^{l}\beta_i\xi_i \tag{6.18}$$

e encontrado o ponto de sela de $\boldsymbol{L}(\mathbf{w}, b, \xi, \alpha, \beta)$. Os parâmetros devem satisfazer as seguintes condições (Equações (6.19) a (6.21)):

$$\frac{\partial \boldsymbol{L}(\mathbf{w}, b, \xi, \alpha, \beta)}{\partial \mathbf{w}} = \mathbf{w} - \sum_{i=1}^{l}\alpha_i y_i \mathbf{z}_i = 0 \tag{6.19}$$

$$\frac{\partial \boldsymbol{L}(\mathbf{w}, b, \xi, \alpha, \beta)}{\partial b} = -\sum_{i=1}^{l}\alpha_i y_i = 0 \tag{6.20}$$

$$\frac{\partial \boldsymbol{L}(\mathbf{w}, b, \xi, \alpha, \beta)}{\partial \xi_i} = s_i C - \alpha_i - \beta_i = 0 \tag{6.21}$$

Aplicando essas condições ao Lagrangiano (Equação (6.18)), o problema (Equação (6.17)) pode ser transformado na Equação (6.22).

$$\begin{aligned} \max\ W(\alpha) &= \sum_{i=1}^{l} \alpha_i - \frac{1}{2}\sum_{i=1}^{l}\sum_{j=1}^{l} \alpha_i \alpha_j y_i y_j K(\mathbf{x}_i, \mathbf{x}_j) \\ sujeito\ a\ &\sum_{i=1}^{l} y_i \alpha_i = 0, \quad 0 \le \alpha_i \le s_i C, \quad i = 1, \ldots, l \end{aligned} \tag{6.22}$$

e as condições de Khun-Tucker são definidas como as Equações (6.23) e (6.24).

$$\bar{\alpha}_i \left(y_i \left(\bar{\mathbf{w}} \cdot \mathbf{z}_i + \bar{b} \right) - 1 + \bar{\xi}_i \right) = 0, \quad i = 1, \ldots, l \tag{6.23}$$

$$(s_i C - \bar{\alpha}_i)\bar{\xi}_i = 0, \quad i = 1, \ldots, l \tag{6.24}$$

O ponto $\mathbf{x}_i$ com o $\bar{\alpha}_i > 0$ correspondente é chamado de vetor de suporte. Existem dois tipos de vetores de suporte, um que corresponde a $0 \le \bar{\alpha}_i \le s_i C$ que se localiza na margem do hiperplano, e outro correspondendo a $\bar{\alpha}_i = s_i C$ que é os vetores classificados incorretamente. Uma importante diferença entre o SVM e o FSVM é que pontos com o mesmo valor de $\bar{\alpha}_i$ podem indicar diferentes tipos de vetores de suporte no FSVM devido ao fator s_i.

O único parâmetro livre C no SVM controla o equilíbrio entre a maximização da margem e a quantidade de classificações incorretas. Um C maior faz o treinamento do SVM ter menos classificações incorretas e uma margem mais estreita. O decréscimo de C faz com que o SVM ignore mais pontos de treinamento e obtenha uma margem mais larga.

No FSVM, pode-se configurar C para ter um valor suficientemente grande. Da mesma forma que o SVM, o sistema obterá margens estreitas e permitirá uma quantidade menor de classificações incorretas se todos os $s_i = 1$. Com diferentes valores de s_i, pode-se controlar o equilíbrio entre os respectivos pontos de treinamento $\mathbf{x}_i$ do sistema. Um valor menor de s_i torna o ponto $\mathbf{x}_i$ correspondente menos importante para o treinamento. Existe somente um parâmetro livre no SVM enquanto o número de parâmetros livres no FSVM é equivalente ao número de pontos de treinamento.

6.4.1. Geração das funções de pertinência fuzzy

Para escolher os graus de pertinência *fuzzy* apropriados para um determinado problema deve-se primeiramente definir os limites inferiores dos graus de pertinência *fuzzy*, e posteriormente deve-se selecionar a propriedade principal do conjunto de dados, e fazer conexões entre essa propriedade e os graus de pertinência *fuzzy*. Realizando a aprendizagem sequencial, primeiramente escolhe-se $\sigma > 0$ como o limite inferior do grau de pertinência *fuzzy*.

Posteriormente, identifica-se que o tempo é a propriedade principal para este tipo de problema, então torna-se o grau de pertinência *fuzzy* s_i função do tempo t_i (Equação (6.25))

$$s_i = f(t_i) \tag{6.25}$$

em que $t_1 \leq \cdots \leq t_l$ e o tempo em que o ponto entrou no sistema. Torna-se o último ponto $\mathbf{x}_l$ o mais importante fazendo $s_l = f(t_l) = 1$, e faz-se o primeiro ponto $\mathbf{x}_1$ o menos importante fazendo $s_1 = f(t_1) = \sigma$. Se é escolhida uma função de pertinência *fuzzy* linear com o tempo, a função se torna a Equação (6.26).

$$s_i = f(t_i) = at_i + b \tag{6.26}$$

Aplicando as condições limite, obtém-se a Equação (6.27).

$$s_i = f(t_i) = \frac{1-\sigma}{t_l - t_1} t_i + \frac{t_l \sigma - t_1}{t_l - t_1} \tag{6.27}$$

Para uma função de pertinência quadrática com o tempo, a função se torna a Equação (6.28).

$$s_i = f(t_i) = a(t_i - b)^2 + c \tag{6.28}$$

Aplicando as condições limites a Equação (6.28), obtém-se a Equação (6.29).

$$s_i = f(t_i) = (1-\sigma)\left(\frac{t_i - t_1}{t_l - t_1}\right)^2 + \sigma \tag{6.29}$$

6.4.2. Dados com propriedades temporais

Métodos de aprendizado e inferências sequenciais são importantes em diversas aplicações envolvendo processamento de sinais em tempo real. Por exemplo, o objetivo é a obtenção de uma máquina de aprendizado que considere mais importantes pontos recentes do que pontos passados. Com esse propósito, pode se selecionar a função de pertinência *fuzzy* como função do tempo que o ponto foi gerado.

Considerando uma sequência de dados de treinamento (Equação (6.30)),

$$(y_1, \mathbf{x}_1, s_1, t_1), \cdots, (y_l, \mathbf{x}_l, s_l, t_l) \tag{6.30}$$

em que $t_1 \leq \cdots \leq t_l$ é o tempo que o ponto chegou ao sistema. Seja a função de pertinência *fuzzy* s_i função do tempo t_i.

O FSVM torna-se capaz de classificar com maior precisão os dados que possuem maior peso de acordo com o grau de pertinência *fuzzy*.

6.4.3. Duas classes com diferentes pesos

O objetivo dessa modificação é proporcionar a precisão necessária para a classificação de uma classe. Por exemplo, dado um ponto, se a máquina de vetores acusa o valor 1, esse ponto faz parte dessa classe com uma precisão muito alta, porém se a máquina de vetores acusa o valor -1, o ponto faz parte da classe com menor precisão ou realmente faz parte de outra classe. Com esse propósito, pode-se selecionar a função de pertinência *fuzzy* como uma função da respectiva classe. Considerando uma sequência de dados de treinamento (Equação (6.31)).

$$(y_1, \mathbf{x}_1, s_1), \cdots, (y_l, \mathbf{x}_l, s_l) \tag{6.31}$$

Seja a função de pertinência s_i uma função da classe y_i (Equação (6.32)).

$$s_i = \begin{cases} s_+, & \text{se} \quad y_i = 1, \\ s_-, & \text{se } y_i = -1. \end{cases} \tag{6.32}$$

Neste caso, o FSVM possui vantagens perante o SVM quando separa pontos com maior precisão do que outros, como é desejado em um processo de detecção de falhas.

6.4.4. Redução de efeitos de outliers

Diversos estudos mostram que o SVM é muito sensível a ruídos e *outliers* (GUYON *et al*., 1996; ZHANG, 1999). O FSVM também pode ser aplicado para reduzir os efeitos dos *outliers*. Pode-se configurar a função de pertinência *fuzzy* como função da distância entre o ponto e o centro da classe dos dados. Essa configuração de função de pertinência pode não ser a melhor maneira de resolver este problema, porém é uma proposta válida para pontos que não possuem muita variação dentro das classes. Dado uma sequência de dados de treinamento (Equação (6.31)), denota-se a média da classe +1 como $\mathbf{x}_+$ e a média da classe -1 como $\mathbf{x}_-$. Seja o raio da classe +1 (Equação (6.33))

$$r_+ = \max_{\{\mathbf{x}_i : y=1\}} |\mathbf{x}_+ - \mathbf{x}_i| \tag{6.33}$$

e o raio da classe -1 (Equação (6.34))

$$r_- = \max_{\{\mathbf{x}_i : y=-1\}} |\mathbf{x}_- - \mathbf{x}_i| \tag{6.34}$$

A função de pertinência *fuzzy* s_i se torna função da média e raio de cada classe (Equação (6.35))

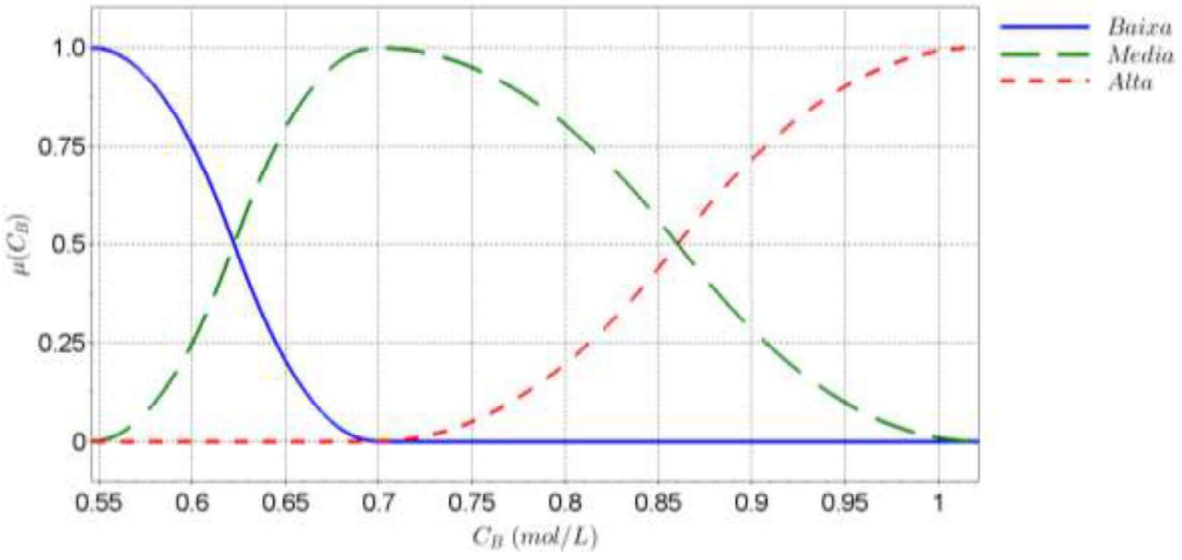
$$s_i = \begin{cases} 1 - |\mathbf{x}_+ - \mathbf{x}_i|/(r_+ + \delta), & \text{se } y_i = 1, \\ 1 - |\mathbf{x}_- - \mathbf{x}_i|/(r_- + \delta), & \text{se } y_i = -1. \end{cases} \tag{6.35}$$

em que $\delta > 0$ é utilizado para evitar o caso $s_i = 0$.

6.5. Estudo de caso 1 - Processo de Produção de Ciclopentanol

Para o sistema reacional de van der Vusse, apresentado na Seção 3.5, foram criadas funções de pertinência para cada uma das variáveis controladas (C_B e T) e manipuladas (F_f e $\dot{Q}_w$). Para cada uma das variáveis, foram formuladas funções de pertinência *fuzzy*, estabelecendo intervalos para que sejam aplicadas as regras das classes ou falhas a serem classificadas. Da Figura 6.13 a Figura 6.16 apresentam as funções de pertinência *fuzzy* para as variáveis C_B, T, $\dot{F}$ e $\dot{Q}_w$, respectivamente.

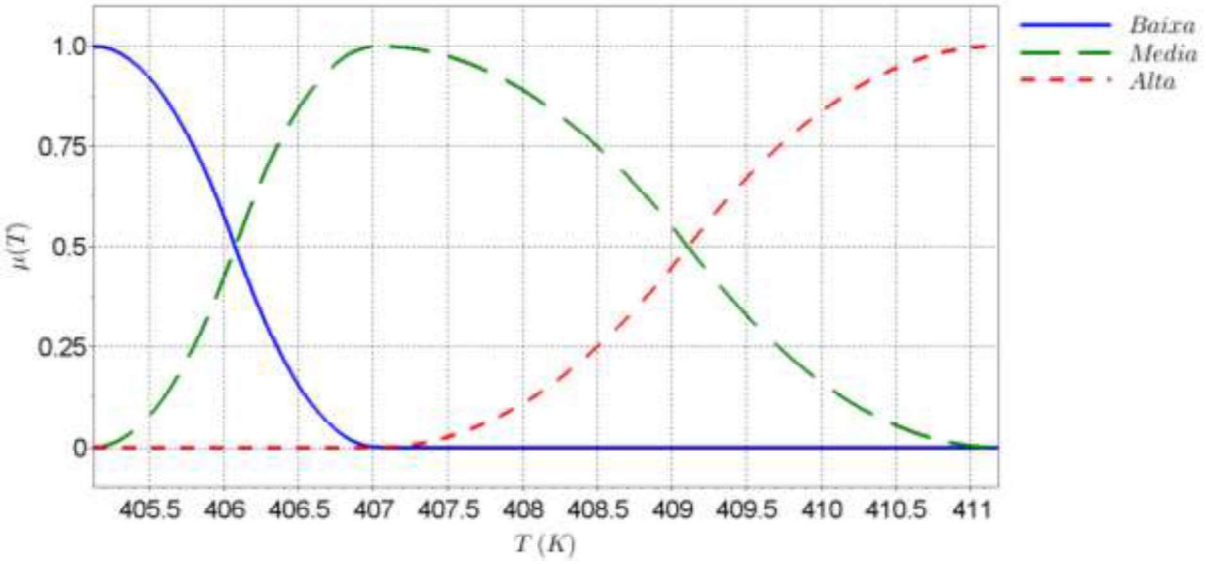
Figura 6.13 - Funções de pertinência *fuzzy* para a variável controlada C_B.



Para todas as variáveis de entrada foram utilizadas funções de pertinências de acordo com os valores mínimos e máximos para cada variável, e de acordo com a média da variável. Foram considerados o valor médio de cada variável, a média do valor dos sinais quando o sistema reacional se encontrava em operação normal. Para cada uma das três funções de pertinência para cada variável, foram estabelecidas variáveis linguísticas – Baixa, para valores

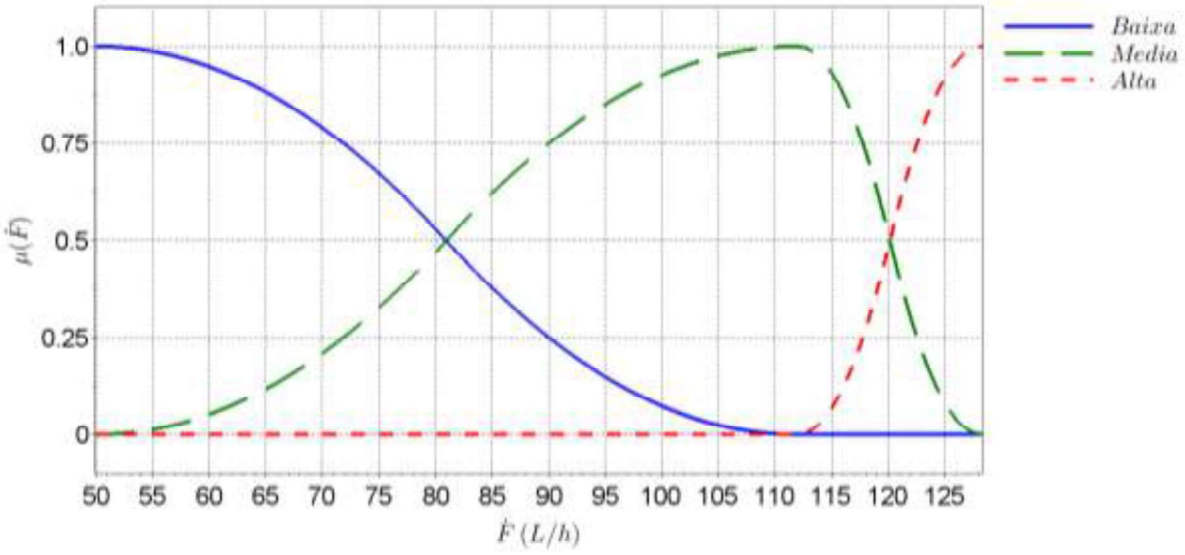
menores que a média da operação normal e ponto de máximo no menor valor da variável; Média, para valores cuja média é o ponto de máximo; Alta, para valores acima da média e ponto de máximo no maior valor da variável.

Figura 6.14 - Funções de pertinência *fuzzy* para a variável controlada *T*.



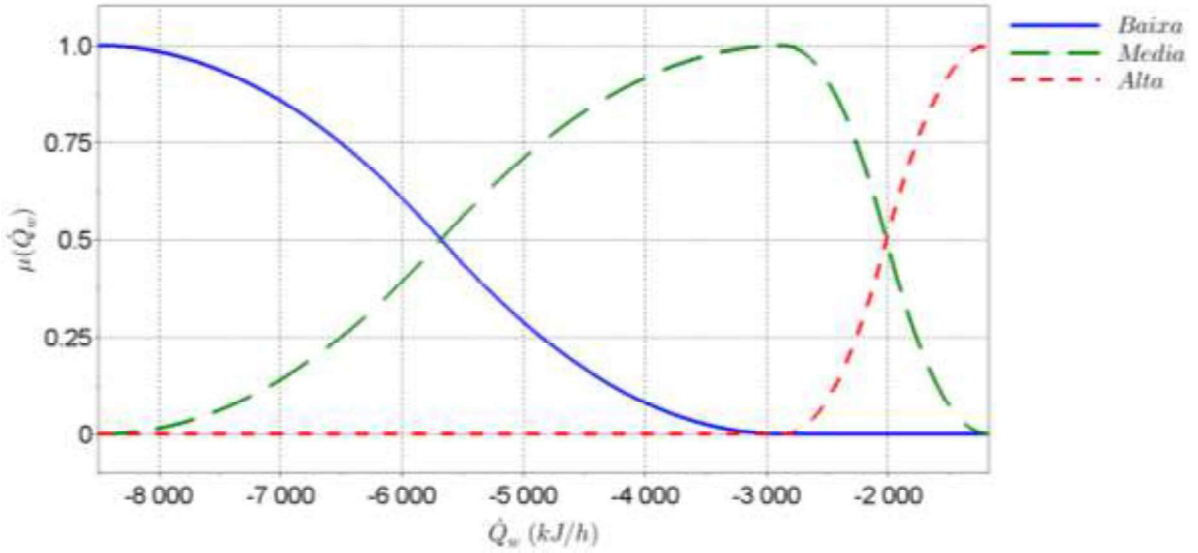
Cada função de pertinência de forma π (duas funções de pertinência de forma s) foi estabelecida de acordo com o valor mínimo, médio e máximo de cada variável, existindo sobreposições entre as funções de pertinência, de maneira que estas sejam complementares, duas a duas.

Figura 6.15 - Funções de pertinência *fuzzy* para a variável manipulada F_f.



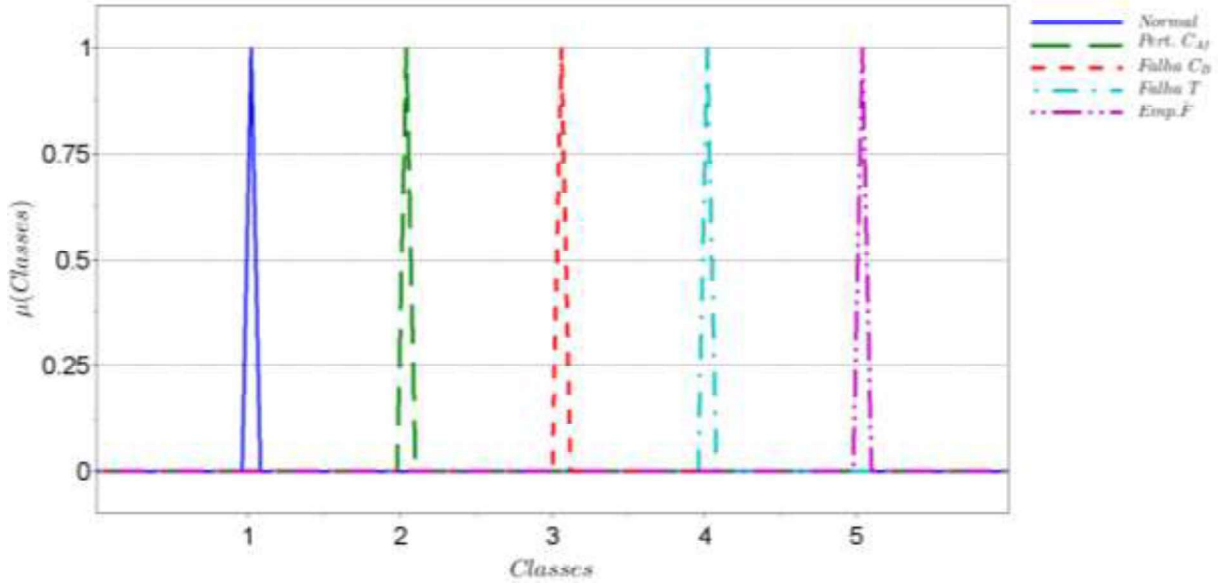
Para a saída do sistema *fuzzy*, foram estabelecidas funções de pertinência de acordo com a classe a que pertencem os dados de entrada, sendo estabelecidas 5 classes (Normal, Pert. C_{Af}, Falha C_B, Falha T e Emp. F), com valores constantes em cada um dos índices das operações.

Figura 6.16 - Funções de pertinência *fuzzy* para a variável manipulada Q_w.



A Figura 6.17 apresenta as funções de pertinência da variável de saída do sistema de detecção de falhas através de lógica *fuzzy*.

Figura 6.17 - Funções de pertinência para a variável de saída Classes.



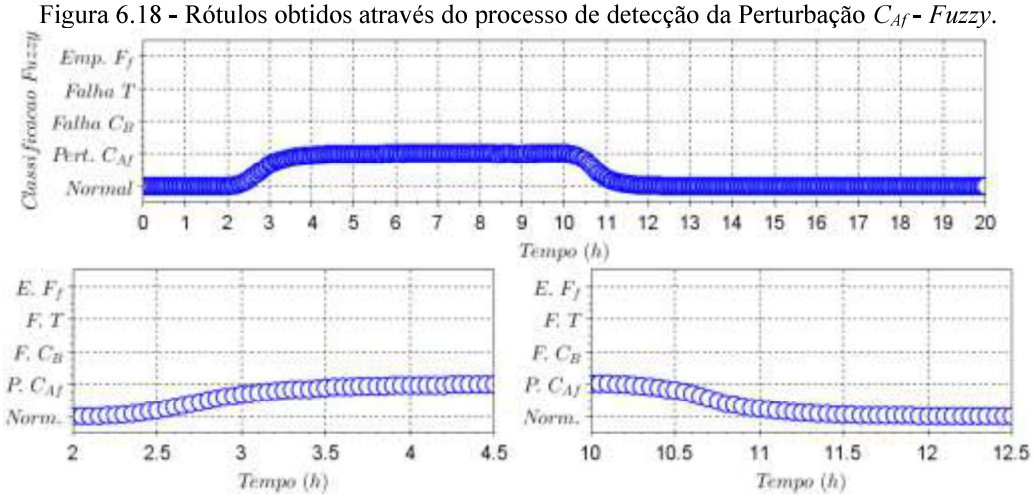
Definidas as funções de pertinência, foram estabelecidas as regras *fuzzy* para a "fuzzyficação" dos dados. As regras que melhor representaram os dados de treinamento (operação normal e falhas) foram as seguintes:

- SE C_B = Média E T = Média E F = Média E Q_w = Média ENTÃO Classes = 1;
- SE C_B = Baixa E T = Baixa E F = Alta ENTÃO Classes = 2;
- SE C_B = Alta E F = Baixa E Q_w = Alta ENTÃO Classes = 3;
- SE T = Alta E F = Baixa E Q_w = Baixa ENTÃO Classes = 4;
- SE C_B = Baixa E T = Média E F = Baixa E Q_w = Alta ENTÃO Classes = 5.

Após o processo de obtenção das funções de pertinência de entrada e saída do modelo de detecção e diagnóstico de falhas através de lógica difusa, é possível verificar através do procedimento de "fuzzyficação" - inferência – "desfuzzyficação", a classe a que cada dado verificado pertence.

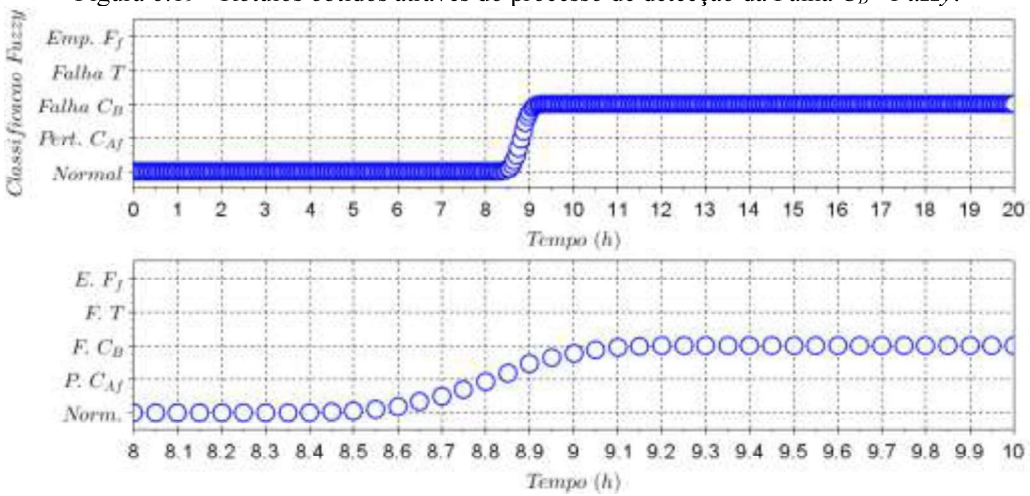
A Figura 6.18 apresenta os resultados do procedimento de detecção e diagnóstico de falhas através da lógica difusa para a perturbação em C_{Af} aplicada no instante 2 *h* e removida no instante 10 *h*. O processo de "desfuzzyficação", devido a dinâmica do sistema, apresentou de forma contínua a modificação da operação normal para a operação em que o sistema se encontra sob a perturbação em C_{Af}. Devido a lógica difusa ser contínua, ou seja, os graus de pertinência possuírem sobreposição, os instantes de transição entre as operações são apresentados de maneira contínua, como apresentado na Figura 6.18.

Figura 6.18 - Rótulos obtidos através do processo de detecção da Perturbação C_{Af} - *Fuzzy*.



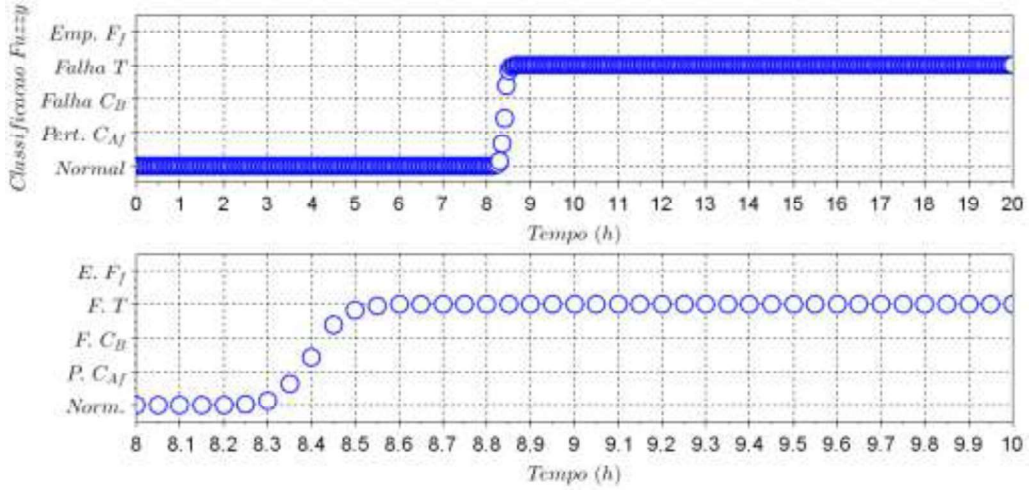
A Figura 6.19 apresenta os resultados do procedimento de detecção e diagnóstico de falhas através da lógica difusa para a Falha C_B aplicada no instante 8 *h*. O sistema de detecção passou a detectar a modificação da operação após 10 intervalos de amostragem (0,50 *h*), devido a dinâmica da falha e o tempo em que o sinal se estabiliza para satisfazer as regras *fuzzy*. A lógica difusa realiza o processo de detecção de falhas de maneira contínua de acordo com as dinâmicas das variáveis classificadas. O sistema de detecção levou 10 intervalos de amostragem (0,5 *h*) para confirmar a detecção da Falha C_B.

Figura 6.19 - Rótulos obtidos através do processo de detecção da Falha C_B - *Fuzzy*.



A Figura 6.20 apresenta os resultados do procedimento de detecção e diagnóstico de falhas através da lógica difusa para a Falha *T* aplicada no instante 8 *h*. O sistema de detecção e diagnóstico passou a detectar a modificação da operação após 3 intervalos de amostragem (0,15 *h*), e devido a dinâmica da falha e o tempo em que o sinal se estabiliza para satisfazer as regras *fuzzy*, o sistema de detecção e diagnóstico detecta a Falha *T* após 5 intervalos de amostragem (0,25 *h*).

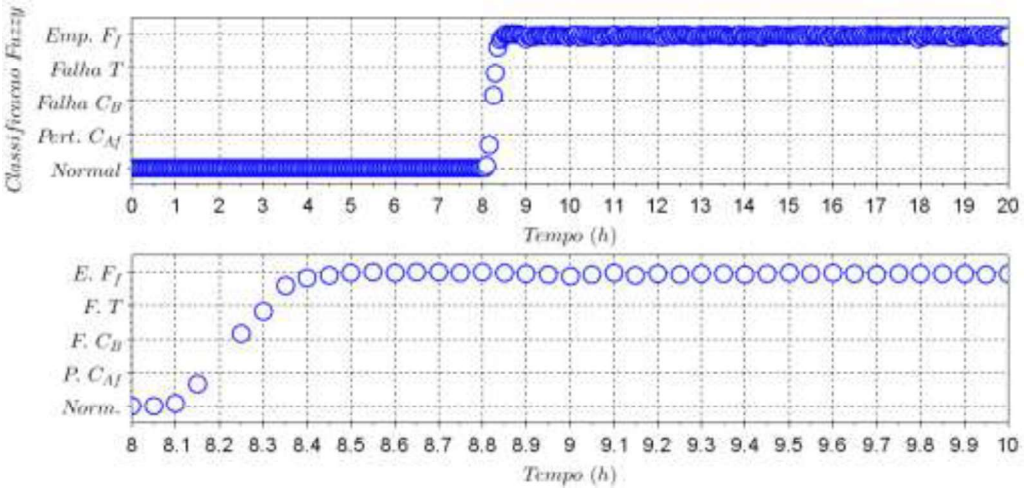
Figura 6.20 - Rótulos obtidos através do processo de detecção da Falha *T* - *Fuzzy*.



A Figura 6.21 apresenta os resultados do procedimento de detecção e diagnóstico de falhas através da lógica difusa para o Emperramento em *F* aplicada no instante 8 *h*. O sistema de detecção passou a modificar a operação de detecção após 1 intervalo de amostragem (0,05 *h*), e devido a dinâmica da falha e o tempo em que o sinal se estabiliza para satisfazer as regras

fuzzy, o sistema de detecção e diagnóstico detecta o Emperramento *F* após 6 intervalos de amostragem (0,3 *h*).

Figura 6.21 - Rótulos obtidos através do processo de detecção da Emperramento *F* - *Fuzzy*.



A Tabela 6.1 apresenta os índices das Equações (3.1) e (3.2) encontrados para o sistema de detecção e diagnóstico através da lógica *fuzzy*.

Tabela 6.1 - Índices encontrados - *Fuzzy*.

Oper.	Índices - *Fuzzy*				
	DF/F	DF/N	DN/N	DN/F	DF_i/F_j
Pert. C_{Af}	0,75	0,08334	0,8334	0,08326	0,25
Falha C_B	0,9167	0,00	1,00	0,04167	0,04167
Falha *T*	0,9625	0,00	1,00	0,0125	0,025
Emp. *F*	0,97083	0,00	1,00	0,004167	0,025

Comparando o método de detecção e diagnóstico de falhas utilizando lógica difusa (*fuzzy*) com os outros métodos previamente apresentados de classificação, o presente método é o único que utiliza variáveis qualitativas para realizar o procedimento, como apresentado na Figuras Figura 6.13 a Figura 6.17. Além disso, dos procedimentos de diagnóstico, ele é também o único estudado que apresenta de maneira contínua a detecção.

6.5.1. Discussão

A utilização de variáveis linguísticas e a possibilidade de geração de funções de pertinência de acordo com os "movimentos" dos dados de treinamento faz com que a lógica difusa seja um caminho válido para a detecção e diagnóstico de falhas. Uma vez que a lógica difusa possui um caráter contínuo, a detecção e diagnóstico de falhas pode ocorrer suavemente. Como pode-se visualizar da Figura 6.18 à Figura 6.21, a detecção ocorre de maneira contínua,

sendo possível a criação de regras que preveem o comportamento do sistema devido a mudanças em cada uma das variáveis.

6.6. Estudo de caso 2 - Planta Química

Para o sistema de Stewart, apresentado na Seção 3.6, foram criadas funções de pertinência para cada uma das variáveis controladas controladas, H_1, T_1, H_2, T_2, H_3 e T_3, e variáveis manipuladas, F_{f1}, Q_1, F_{f2}, Q_2, F_R e Q_3. Para cada uma das variáveis, foram formuladas funções de pertinência *fuzzy*, estabelecendo intervalos para que sejam aplicadas as regras das classes ou falhas a serem classificadas.

No presente estudo de caso, como primeira opção, procurou-se utilizar para todas as funções de pertinência de todas as variáveis controladas ou manipuladas funções do tipo sino. Para verificar a adequação do modelo obtido através de funções de pertinência do tipo sino, foi compilado um segundo modelo em que todas as funções de pertinência eram do tipo triangular, servindo de base para comparação para o primeiro modelo.

No Apêndice A, Seção A.5, estão apresentadas todas as funções de pertinência do formato sino para cada uma das variáveis controladas ou manipuladas, assim como as devidas aplicações dessas quando classificando estados estacionários e operações faltosas, de acordo com a metodologia seguida ao decorrer deste trabalho.

O método *Fuzzy* tem a capacidade de classificar dados de acordo com o procedimento de *fuzzyficação* → funções de pertinência → *defuzzyficação*, como apresentado neste capítulo.

A Tabela 6.2 apresenta os índices obtidos através das equações propostas no capítulo 3, Seção 3.3.

6.6.1. Discussão

Para o estudo de caso 2, a melhor forma encontrada para poder realizar a detecção das falhas foi realizá-las em separado para cada uma das variáveis, ao invés de realiza-las todas em um único processo de detecção de utilizando as 12 variáveis ao mesmo tempo. Quebrando o problema de detecção em 12 problemas menores, o procedimento de criação das funções de pertinência tornou-se mais intuitivo e dinâmico, tendo menos *outliers* e detecções errôneas.

Para algumas das variáveis, as detecções de ambos os estados estacionários ocorreram rapidamente e de maneira precisa, enquanto que para outras variáveis, como para H_x, os estados estacionários foram detectados após quase 5 minutos (ou 10 instantes), um tempo considerado longo, quando o tempo total de simulação é de 60 minutos.

Tabela 6.2 - Índices - Estudo de Caso 2 - Fuzzy Sino (S) e Fuzzy Triangular (T).

Oper.	Índices	Fuzzy S	Fuzzy T	Oper.	Índices	Fuzzy S	Fuzzy T
ee1	DF/F	0,9600	0,9600	*ee2*	DF/F	0,9600	0,9600
	DN/F	0,0400	0,0400		DN/F	0,0400	0,0400
	DN/N	0,9714	0,9714		DN/N	0,9714	0,9714
	DF/N	0,0286	0,0286		DF/N	0,0286	0,0286
$H1_1$	DF/F	1,0000	1,0000	$Ff1_1$	DF/F	1,0000	1,0000
	DN/F	0,0000	0,0000		DN/F	0,0000	0,0000
	DN/N	0,9143	0,9143		DN/N	0,9857	0,9857
	DF/N	0,0857	0,0857		DF/N	0,0143	0,0143
$H1_2$	DF/F	1,0000	1,0000	$Ff1_2$	DF/F	1,0000	1,0000
	DN/F	0,0000	0,0000		DN/F	0,0000	0,0000
	DN/N	0,8600	0,8600		DN/N	0,9714	0,9714
	DF/N	0,1400	0,1400		DF/N	0,0286	0,0286
$H2_1$	DF/F	1,0000	1,0000	$Ff2_1$	DF/F	1,0000	1,0000
	DN/F	0,0000	0,0000		DN/F	0,0000	0,0000
	DN/N	0,8400	0,8400		DN/N	0,9857	0,9857
	DF/N	0,1600	0,1600		DF/N	0,0143	0,0143
$H2_2$	DF/F	1,0000	1,0000	$Ff2_2$	DF/F	1,0000	1,0000
	DN/F	0,0000	0,0000		DN/F	0,0000	0,0000
	DN/N	0,8400	0,8400		DN/N	0,9857	0,9857
	DF/N	0,1600	0,1600		DF/N	0,0143	0,0143
$H3_1$	DF/F	1,0000	1,0000	F_{R1}	DF/F	1,0000	1,0000
	DN/F	0,0000	0,0000		DN/F	0,0000	0,0000
	DN/N	0,7800	0,7800		DN/N	0,9857	0,9857
	DF/N	0,2200	0,2200		DF/N	0,0143	0,0143
$H3_2$	DF/F	1,0000	1,0000	F_{R2}	DF/F	1,0000	1,0000
	DN/F	0,0000	0,0000		DN/F	0,0000	0,0000
	DN/N	0,7800	0,7800		DN/N	0,9714	0,9714
	DF/N	0,2200	0,2200		DF/N	0,0286	0,0286
$T1_1$	DF/F	1,0000	1,0000	$Q1_1$	DF/F	1,0000	1,0000
	DN/F	0,0000	0,0000		DN/F	0,0000	0,0000
	DN/N	0,9800	0,9800		DN/N	0,9600	0,9600
	DF/N	0,0200	0,0200		DF/N	0,0400	0,0400
$T1_2$	DF/F	1,0000	1,0000	$Q1_2$	DF/F	1,0000	1,0000
	DN/F	0,0000	0,0000		DN/F	0,0000	0,0000
	DN/N	0,9400	0,9400		DN/N	0,9800	0,9800
	DF/N	0,0600	0,0600		DF/N	0,0200	0,0200
$T2_1$	DF/F	1,0000	1,0000	$Q2_1$	DF/F	1,0000	1,0000
	DN/F	0,0000	0,0000		DN/F	0,0000	0,0000
	DN/N	0,9800	0,9800		DN/N	0,9600	0,9600
	DF/N	0,0200	0,0200		DF/N	0,0400	0,0400
$T2_2$	DF/F	1,0000	1,0000	$Q2_2$	DF/F	1,0000	1,0000
	DN/F	0,0000	0,0000		DN/F	0,0000	0,0000
	DN/N	0,9400	0,9400		DN/N	0,9800	0,9800
	DF/N	0,0600	0,0600		DF/N	0,0200	0,0200
$T3_1$	DF/F	1,0000	1,0000	$Q3_1$	DF/F	1,0000	1,0000
	DN/F	0,0000	0,0000		DN/F	0,0000	0,0000
	DN/N	0,9600	0,9600		DN/N	0,9800	0,9800
	DF/N	0,0400	0,0400		DF/N	0,0200	0,0200
$T3_2$	DF/F	1,0000	1,0000	$Q3_2$	DF/F	1,0000	1,0000
	DN/F	0,0000	0,0000		DN/F	0,0000	0,0000
	DN/N	0,9000	0,9000		DN/N	0,9800	0,9800
	DF/N	0,1000	0,1000		DF/N	0,0200	0,0200

Os resultados obtidos pela detecção de falhas utilizando o modelo *Fuzzy* está apresentado no Apêndice A da Figura A.105 até a Figura A.141.

6.7. Comentários

A Lógica *Fuzzy* quando utilizada como metodologia de detecção de falhas possui vantagens perante outras metodologias como não necessitar de um procedimento de treinamento como os métodos utilizando redes neuronais e máquinas de vetores de suporte, além de ser capaz de detectar as falhas de maneira contínua, não apresentando descontinuidades nos dados de detecção, entretanto, existe uma dificuldade na obtenção das melhores funções de pertinência para determinado sistema de detecção de falhas. Os resultados obtidos através da Lógica *Fuzzy* foram compatíveis com os outros métodos de detecção.

No próximo capítulo, é apresentada a proposta de teoria de detecção de falhas com aprendizado profundo, suas aplicações em detecção e diagnóstico de falhas, e os resultados da detecção de dados para os estudos de casos.

CAPÍTULO 7
DETECÇÃO DE FALHAS COM APRENDIZADO PROFUNDO

7.1. Introdução

A aprendizagem profunda (*deep learning* ou aprendizado de máquina profundo) é uma técnica de aprendizado de máquina que ensina os computadores a "aprenderem" de forma parecida com os humanos: aprender com exemplos. O aprendizado profundo é uma tecnologia chave para diversas novas tecnologias, como carros autônomos, reconhecimento de faces, imagens e voz, reconhecimento de padrões de comportamentos, suporte técnico personalizado, detecção de depósitos minerais através de imagens de satélites, classificação de doenças oculares, diagnóstico de câncer, entre outras.

A aprendizagem profunda (*deep learning*, DL), de acordo com Deng e Yu (2013) é definida como:

- Classe de técnicas de *Machine Learning* (ML) que exploram muitas camadas de processamento de informações não lineares para extração e transformação de características supervisionadas ou não-supervisionadas, e para análise e classificação de padrões;
- Sub-campo dentro do ML que é baseado em algoritmos para aprender múltiplos níveis de características (*features*), a fim de modelar relações complexas entre os dados. Recursos e conceitos de nível superior são assim definidos em termos de níveis mais baixos, e essa hierarquia de recursos é chamada de arquitetura profunda. A maioria desses modelos é baseada em aprendizado não-supervisionado de dados;
- Sub-campo de ML que é baseado no aprendizado de vários níveis de representações, correspondendo a uma hierarquia de recursos ou fatores ou conceitos, em que conceitos de nível superior são definidos a partir de conceitos de nível inferior, e os mesmos conceitos de nível inferior podem ajudar a definir muitos conceitos de nível superior. A aprendizagem profunda faz parte de uma família mais ampla de métodos de aprendizado de máquina baseados em representações de aprendizagem. Uma amostra (por exemplo, uma imagem) pode ser representada de várias maneiras (por exemplo, um vetor de pixels), mas algumas representações facilitam o aprendizado de tarefas de interesse (por

exemplo, diferenciar imagens de rostos humanos) a partir de exemplos, e pesquisa nesta área tenta definir que técnicas obtêm melhores representações e como aprendê-las;

- Aprendizagem profunda é um conjunto de algoritmos em ML que tenta aprender em múltiplos níveis, correspondendo a diferentes níveis de abstração. Normalmente são utilizadas RNAs. Os níveis nesses modelos estatísticos aprendidos correspondem a níveis distintos de características, onde características de nível mais alto são definidos a partir das características de nível inferior, e essas características inferiores ajudam na definição de diversas características superiores;
- Aprendizado profundo é uma nova área de pesquisa em ML, que foi introduzida com o objetivo de aproximar o aprendizado de máquina de um dos seus objetivos originais: Inteligência Artificial. O DL consiste em aprender múltiplos níveis de representação e abstração que ajudam a compreender dados como imagens, sons e texto.

Com o principal objetivo de automaticamente obter informações e características inerentes a grupos e subgrupos de dados, o aprendizado profundo apresenta uma área muito promissora para a detecção e diagnóstico de falhas em sistemas de controle de processos químicos, com a capacidade de identificar automaticamente características que seriam capazes de separar e classificar entradas e saídas de processos, sejam estas relativas às operações normal e anormais do sistema em questão.

Nesse capítulo, serão propostos algoritmos de aprendizado de máquina *deep learning* para a detecção e diagnóstico de falhas, cujos resultados dos procedimentos de detecção para o estudo de caso serão comparados neste capítulo com os outros métodos apresentados anteriormente neste trabalho.

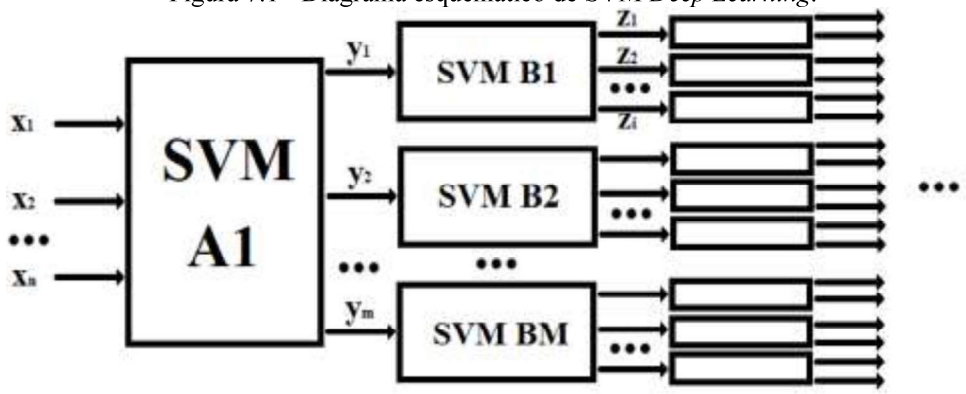
7.2. *Deep Learning* SVM

Para se utilizar o SVM com aprendizagem profunda, torna-se necessário utilizar diversas camadas de processamento para a classificação de informações, a partir de características mais simples até características mais complexas. Por exemplo, através de uma máquina de vetores de suporte, capaz de classificar *n* características, torna-se possível uma série de *n* novas máquinas de vetores de suporte, cada uma delas capaz de classificar inúmeras novas características, criando uma cascata de máquinas de vetores de suporte de complexidades

diversas, porém mais simples do que uma única máquina de vetor capaz de separar uma quantidade excessiva de informações (Figura 7.1).

Outro método de utilização de máquinas de vetores de suporte com aprendizado profundo é a utilização de diferentes máquinas de vetores de suporte para diferentes variáveis de entrada do sistema de detecção de falhas, classificando ou separando características importantes relativas a cada variável, combinando posteriormente essas características para classificar os dados de acordo com o comportamento em conjunto dessas variáveis.

Figura 7.1 - Diagrama esquemático de SVM *Deep Learning*.



Cada máquina de vetores de suporte no procedimento de aprendizado profundo pode ser configurada de acordo com o objetivo desejado em cada camada de separação. Caso a característica a ser classificada seja conhecida em cada etapa, as máquinas de vetores de suporte utilizadas podem ser de classificação binária ou máquinas de vetores de suporte de classificação *One-Against-One* ou *One-Against-All*, uma vez da necessidade da construção do vetor de rótulos da saída de cada máquina de vetores de suporte, sendo considerado um algoritmo de aprendizado supervisionado. Caso não sejam conhecidas as características, existe a possibilidade de utilização da SVM de regressão, separando os dados de acordo com a proximidade do hiperplano obtido no treinamento para cada máquina de vetores de suporte. No treinamento, os dados separados em cada camada continuam respectivamente para a máquina de vetores de suporte da camada seguinte referente a primeira classificação, e assim por diante.

O algoritmo seria capaz então de decidir, através de funções como *softmax*, em que são selecionados os dados com maior valor de rótulo para cada dado classificado, a qual categoria ou classe que determinada amostra pertence.

7.3. *Deep Learning* ANN

O problema fundamental de atribuição de crédito (MINSKY, 1963) estuda a respeito de que componentes modificáveis em um sistema de aprendizagem são responsáveis pelo seu

sucesso ou falha, e quais modificações podem melhorar o desempenho desses sistemas. Existem diversos métodos genéricos de atribuição de crédito para solucionadores de problemas universais, porém um dos mais importantes são os métodos relativos a aprendizagem profunda (*Deep Learning*, DL) em redes neuronais artificiais (*Artificial Neural Networks*, ANN).

Uma rede neuronal padrão consiste em diversos processadores simples e interconectados chamados neurônios, cada um produzindo uma sequência de ativações de valor real (GALUSHKIN, 2007). Os neurônios de entrada são ativados através de sensores que percebem o ambiente ou sistema, outros neurônios são ativados através de conexões ponderadas dos neurônios ativados anteriormente. Alguns neurônios podem influenciar o ambiente ou sistema iniciando ações. A atribuição de crédito ou aprendizado é o processo de procura dos pesos das conexões que fazem a rede neuronal exibir o comportamento desejado, tal qual dirigir um carro ou reconhecer um padrão. Dependendo do problema e como os neurônios são conectados, tal comportamento pode requerer longas cadeias causais de estágios computacionais, em que cada estágio transforma (muitas vezes de maneira não linear) a ativação agregada da rede. A aprendizagem profunda é sobre atribuir crédito de forma precisa ao longo de diversos estágios computacionais como esses.

Como visto anteriormente, modelos de redes neuronais ditos “rasos” (*shallow*) possuem poucos estágios e estiveram disponíveis por muitas décadas. Modelos não-lineares com sucessivas camadas de neurônios datam pelo menos da década de 1960 e 1970 (JOSEPH, 1961; IVAKHNENKO, 1968, 1971; IVAKHNENKO; LAPA, 1965; VIGLIONE, 1970). Um método eficiente de decréscimo de gradiente para aprendizado supervisionado em redes discretas e diferenciais de profundidade arbitrária chamado *backpropagation* (retroalimentação) foi aplicado em redes neuronais artificiais em 1981 (WERBOS, 1981, 2006; LECUN, 1985, 1988; PARKER, 1985). A popularização da retroalimentação para redes neuronais veio através de Rumelhart e colaboradores (1986), experimentalmente demonstrando o surgimento de representações internas úteis em camadas ocultas.

O treinamento de redes neuronais artificiais de aprendizagem profunda com retroalimentação se provou muito difícil na prática até o final da década de 1980 (ELMAN, 1990; JORDAN, 1986, 1997) e se tornou muito pesquisado no início da década de 1990. O aprendizado profundo tornou-se praticamente viável com a ajuda da aprendizagem não-supervisionada (HOCHREITER, 1991). Nas décadas de 1990 e 2000 existiram também muitos aperfeiçoamentos em aprendizado profundo supervisionado. A partir da década de 2000, as Redes Neuronais Profundas (*Deep Neural Networks*, DNN) atraíram finalmente atenção generalizada, principalmente por superar métodos alternativos de aprendizado de máquina, como as máquinas de *kernel* (SVM) (SCHÖLKOPF *et al*., 1998; VAPNIK, 1995), em diversas

aplicações importantes. O estudo das máquinas de *kernel* neste trabalho foi realizado com o propósito de comparação entre as diferentes metodologias de detecção de falhas. De fato, desde 2009, redes neuronais profundas supervisionadas venceram diversas competições oficiais internacionais de reconhecimento de padrões e têm se tornado mais relevantes no campo do aprendizado de reforço (*reinforcement learning*, RL) onde não existe um supervisor (RAINA *et al*., 2009; LEE *et al*., 2009a, 2009b).

Redes com topologias de alimentação direta (*feedforward*) e de alimentação recorrente (*feedback*) venceram concursos internacionais de metodologias de classificação de dados, concorrendo com diversas outras metodologias baseadas em aprendizado de máquina. De certo modo, redes recorrentes (cíclicas) são as mais "profundas" de todas as redes neuronais. As redes *feedback*, quando comparadas as redes *feedforward*, são solucionadores mais poderosos e podem criar e processar memórias de sequências arbitrárias de padrões de entrada (SCHMIDHUBER, 1990; SIEGELMANN; SONTAG, 1991). Diferentemente de métodos tradicionais para síntese automática de programas sequenciais (DEVILLE; LAU, 1994; SOLOWAY, 1986), redes recorrentes podem aprender programas que misturam processamento de informações sequenciais e paralelas de maneira natural e eficiente, explorando o paralelismo massivo visto como crucial para sustentar o rápido declínio do custo computacional observado nos últimos 75 anos.

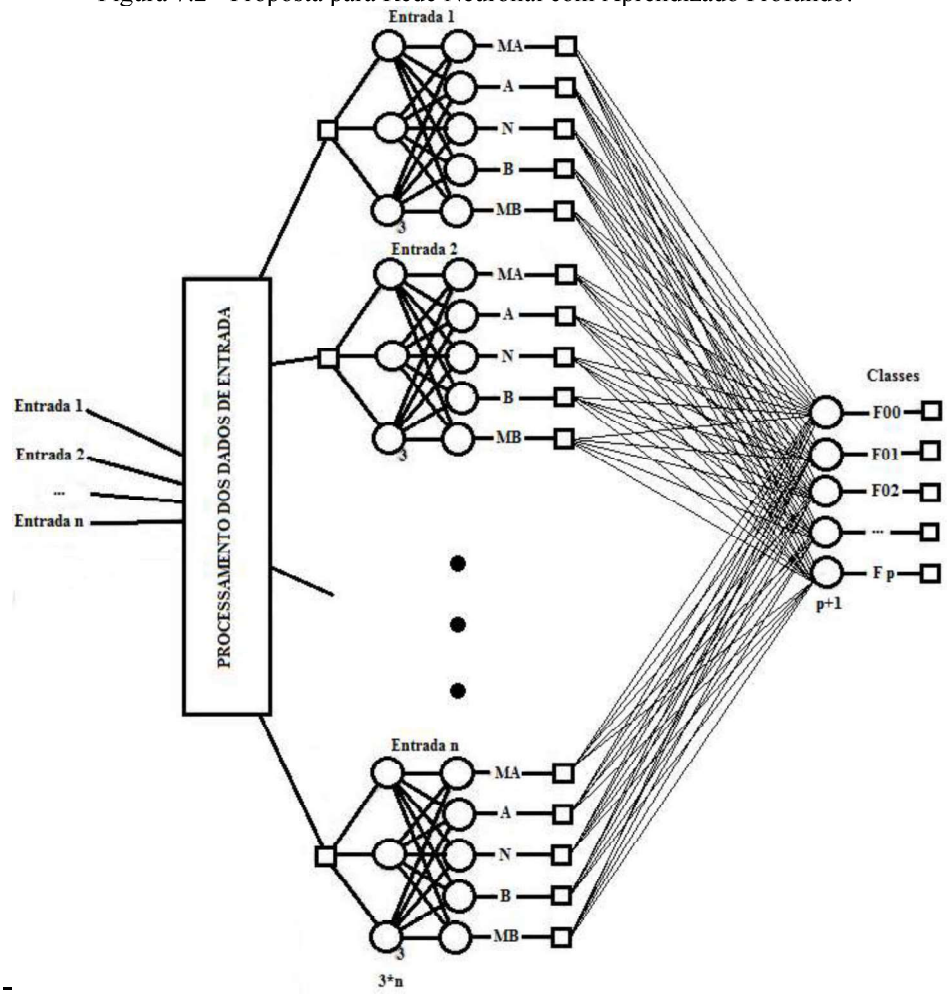
As redes neuronais de aprendizado profundo (DNN) são redes neuronais que normalmente possuem um número grande de camadas ocultas. Comparando o aprendizado profundo com redes neuronais de aprendizado "raso", o erro obtido na fase de treinamento das DNNs é consideravelmente menor do que em redes neuronais com somente uma camada oculta. Redes neuronais simples com alimentação direta possuem a capacidade de aproximar qualquer função, ou seja, aprender qualquer informação. Porém, as redes neuronais "rasas" podem necessitar mais neurônios que uma rede profunda, a arquitetura "rasa" pode não se adequar aos tipos de problemas que necessitam ser solucionados, ou então as redes "rasas" são mais difíceis de treinar com os algoritmos atuais. Atualmente, as redes profundas possuem excelentes resultados para detecção de padrões, com mais de cem camadas ocultas (BA; CARUANA, 2014).

7.4. Redes Neuronais Linguísticas

A proposta de redes neuronais com aprendizado profundo utilizando variáveis linguísticas apresentada neste trabalho utiliza técnicas de paralelismo e processos em cascata para realizar o treinamento do modelo de classificação das falhas. A rede proposta foi designada como Redes Neuronais Linguísticas, ou LNN. A Figura 7.2 apresenta a ideia principal

envolvida na proposta. A ideia principal é utilizar a estrutura de funções de pertinência e linguagem natural para as redes neuronais de múltiplas camadas. A camada de entrada da metodologia é formada por cada variável do sistema, então são desenvolvidas *n* (em que *n* é o número de variáveis) redes neuronais com duas camadas cada, sendo a primeira camada formada por três neurônios e a segunda camada formada por cinco neurônios. As redes neuronais de duas camadas são treinadas com os valores de cada variável, de forma que existam cinco variáveis linguísticas na saída das redes (MA, muito alto; A, alto; N, neutro ou médio; B, baixo; e MB, muito baixo). As saídas dessas redes se encontram na camada de saída do sistema, fornecendo a probabilidade de que determinada amostra seja classificada nas diferentes *p*+1 classes, sendo uma classe de operação normal, e as outras *p* classes as operações faltosas.

Figura 7.2 - Proposta para Rede Neuronal com Aprendizado Profundo.



7.5. Estudo de Caso - Processo de Produção de Ciclopentanol

Para ilustrar a aplicação da detecção e identificação de falhas através de redes neuronais de aprendizagem profunda, foi utilizado o processo de produção de Ciclopentanol a partir da reação de van der Vusse, explicada anteriormente na Seção 3.5.

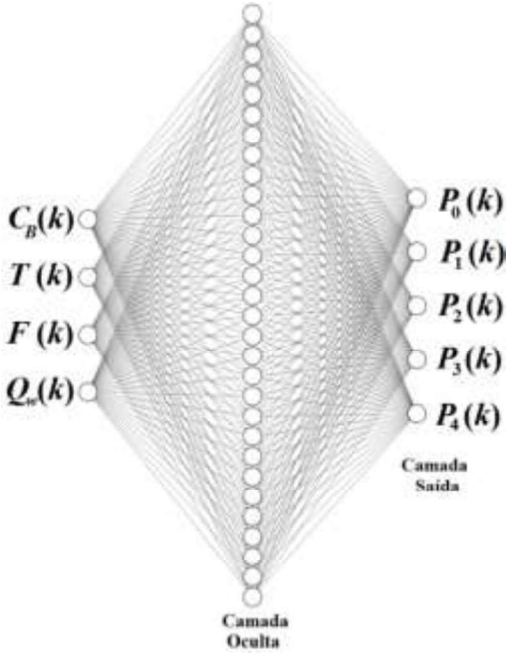
7.5.1. Redes Neuronais Artificiais Supervisionadas com Única Camada

Para as redes neuronais artificiais de única camada oculta (*Shallow Neural Networks*, SNN) foi necessário treinar a rede neuronal com as informações de treinamento de todas as falhas. controladas, C_B e T, e variáveis manipuladas, $\dot{F}$ e $\dot{Q}_w$, como entradas do modelo SNN.

Para a saída do modelo SNN, foram estabelecidas probabilidades para que cada dado faça parte de cada uma das classes, sendo essas classes as operações faltosas (Pert. C_{Af} - 1, Falha C_B - 2, Falha T - 3 e Emperramento F - 4) e a operação normal (0). Para a saída do treinamento e validação, foram estabelecidas juntamente às variáveis de treinamento e validação o vetor de probabilidades $P = [P_0 \quad P_1 \quad P_2 \quad P_3 \quad P_4]^T$, para $P_i = 1$ se i = falha, e $P_j = 0$ se $j \neq$ falha.

A rede então foi definida com a entrada $x = [C_B \quad T \quad \dot{F} \quad \dot{Q}_w]^T$ e saída $y = [P_0 \quad P_1 \quad P_2 \quad P_3 \quad P_4]^T$. A estrutura da rede foi definida com 4 nós na camada de entrada, cada nó para cada uma das variáveis do vetor x, com uma camada interna de neurônios, sendo uma única camada oculta com 30 nós, escolhidos de maneira heurística, e uma camada de saída com 5 nós, cada um dele especificando a probabilidade de determinado dado fazer parte de cada uma das classes (Figura 7.3).

Figura 7.3 - Estrutura da SNN: Entrada-Camada Oculta-Saída.



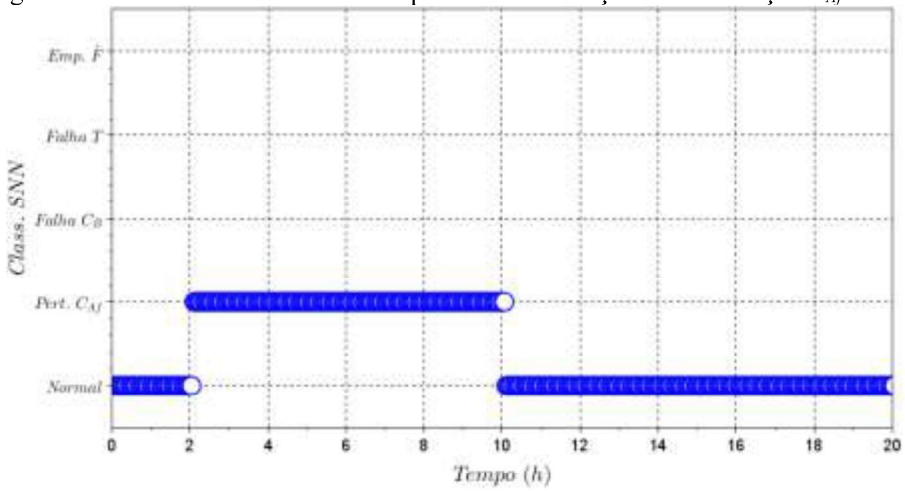
Para o treinamento do sistema de detecção e diagnóstico através de SNN, foram utilizados dois terços dos dados, enquanto o um terço restante foi utilizado para validação. Para o treinamento da rede, foram estabelecidas mil (1000) épocas, de forma a minimizar o MSE entre os dados de validação e os dados de treinamento. Cada época no processo de treinamento

da rede neuronal é utilizada para melhorar a matriz de pesos **W**, de acordo com a Equação (5.23). A taxa de aprendizado utilizado foi de 0,9 e o tempo de treinamento *offline* das 1000 épocas foi de 100 minutos.

Após o treinamento e validação da rede, foi armazenada a matriz peso **W** e as informações pertinentes à rede SNN, como o número de neurônios da camada oculta da rede e a taxa de aprendizado cuja rede foi treinada, que posteriormente foi utilizada para a classificação de novos dados. Para a obtenção da classe dos dados testados foi estabelecida a maior probabilidade de fazer parte de determinada classe para cada instante, e esses dados foram plotados para melhor visualização.

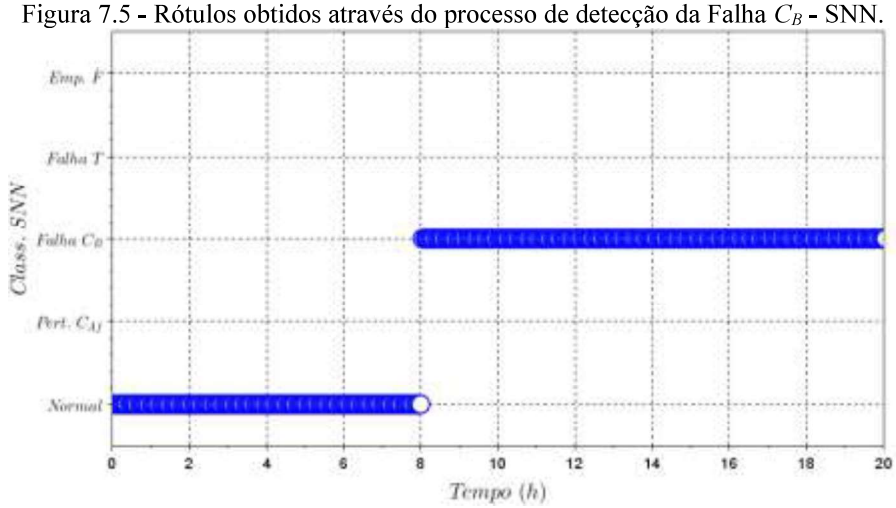
A Figura 7.4 apresenta a classificação dos dados da perturbação em C_{Af}, aplicada no instante de 2 h e removida no instante de 10 h. A perturbação passou a ser detectada após um intervalo de amostragem (0,05 h). Após a remoção da perturbação, o sistema de detecção levou um intervalo de amostragem (0,05 h) para detectar novamente a operação normal.

Figura 7.4 - Rótulos obtidos através do processo de detecção da Perturbação C_{Af} - SNN.



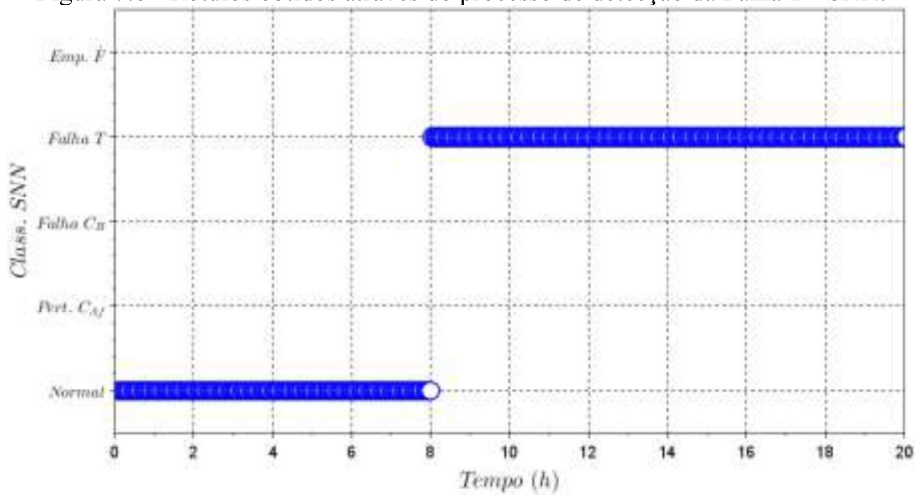
A Figura 7.5 apresenta a classificação dos dados da falha em C_B, aplicada no instante de 8 h. A falha passou a ser detectada após um intervalo de amostragem (0,05 h).

Figura 7.5 - Rótulos obtidos através do processo de detecção da Falha C_B - SNN.



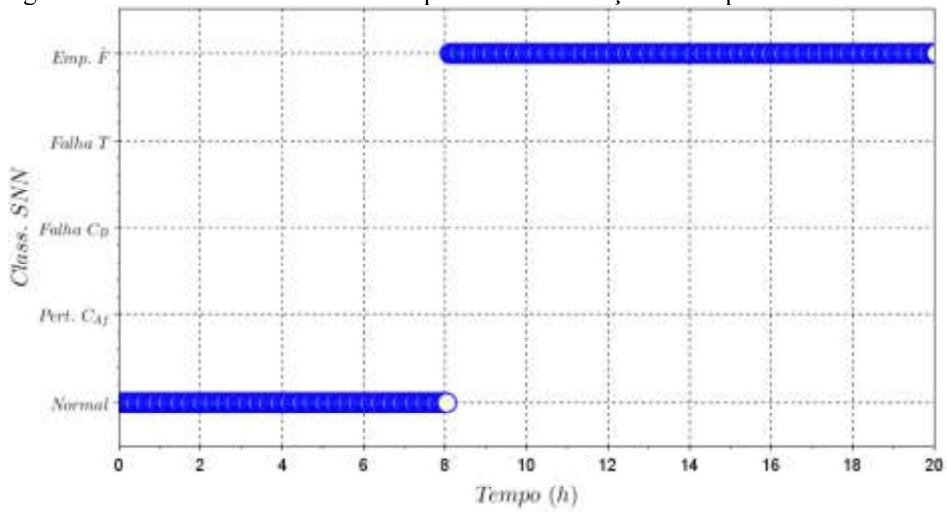
A Figura 7.6 apresenta a classificação dos dados da falha no sensor *T*, aplicada no instante de 8 h. A falha foi detectada após um intervalo de amostragem (0,05 h).

Figura 7.6 - Rótulos obtidos através do processo de detecção da Falha *T* - SNN.



A Figura 7.7 apresenta a classificação dos dados do emperramento em *F*, aplicada no instante de 8 h. A falha foi detectada após um intervalo de amostragem (0,05 h).

Figura 7.7 - Rótulos obtidos através do processo de detecção do Emperramento *F* - SNN.



Foram contabilizados todos os instantes nos quais o comportamento do sistema era normal e todos os instantes em que o sistema estava passando pelas diferentes falhas treinadas, de forma que foram criados índices que mostram qual a porcentagem de identificações corretas ou não, de acordo com as Equação (3.1), para todas as operações com falhas testadas para o sistema de detecção e diagnóstico através de SNN.

A Tabela 7.1 apresenta os índices encontrados para o sistema de detecção e diagnóstico através do SNN.

Tabela 7.1 - Índices encontrados - SNN.

Oper.	Índices - SNN			
	DF/F	DF/N	DN/N	DN/F
Pert. C_{Af}	1,00	0,00	1,00	0,00
Falha C_B	1,00	0,00	1,00	0,00
Falha *T*	1,00	0,00	1,00	0,00
Emp. *F*	1,00	0,00	1,00	0,00

O método de detecção e diagnóstico de falhas através de redes neuronais artificiais com única camada oculta (SNN), comparado aos outros métodos de detecção e diagnóstico de falhas, resulta em detecção e diagnóstico praticamente instantâneos, levando em todos os cenários testados somente um instante para a detecção (apesar de em uma das falhas, a Perturbação C_{Af}, o SNN ter erros de classificação). Porém, o treinamento de uma rede neuronal artificial, além das dificuldades inerentes a obtenção de sua estrutura otimizada, demanda uma alta capacidade computacional, levando em alguns casos mais de 100 minutos para o seu treinamento.

7.5.2. Redes Neuronais Artificiais Supervisionadas com Múltiplas Camadas

Para as redes neuronais artificiais com múltiplas camadas ocultas (*Deep Neural Networks*, DNN) e algoritmo *Back Propagation*, foi necessário treinar a rede neuronal com as

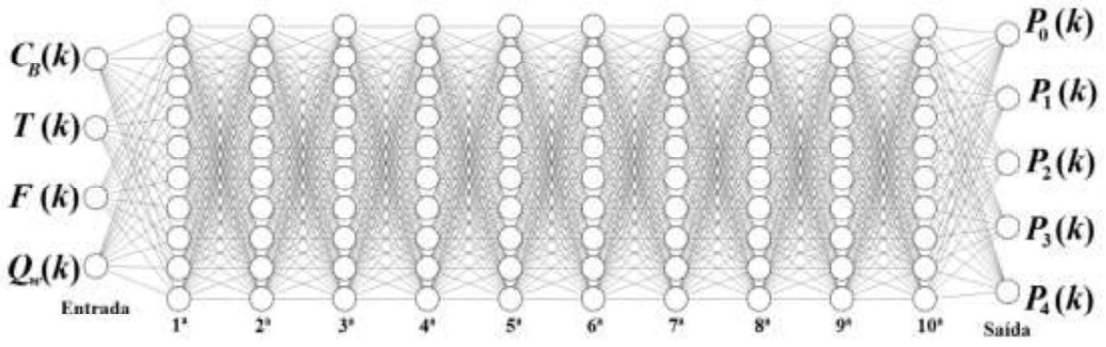
informações de treinamento de todas as falhas utilizando as variáveis controladas, C_B e T, e as variáveis manipuladas, $\dot{F}$ e $\dot{Q}_w$, como entradas do modelo DNN.

Para a saída do modelo DNN, foram estabelecidas probabilidades para que cada dado faça parte de cada uma das classes, sendo essas classes as operações faltosas (Pert. C_{Af} - 1, Falha C_B - 2, Falha T - 3 e Emperramento F - 4) e a operação normal (0). Para a saída do treinamento e validação, foram estabelecidas juntamente às variáveis de treinamento e validação o vetor de probabilidades $P = [P_0 \quad P_1 \quad P_2 \quad P_3 \quad P_4]^T$, para $P_i = 1$ se i = falha, e $P_j = 0$ se $j \neq$ falha.

A rede então foi definida com a entrada $x = [C_B \quad T \quad \dot{F} \quad \dot{Q}_w]^T$ e saída $y = [P_0 \quad P_1 \quad P_2 \quad P_3 \quad P_4]^T$. A estrutura da rede foi definida com 4 nós na camada de entrada, cada nó para cada uma das variáveis do vetor x, com uma camada interna de neurônios, sendo dez camadas oculta com 10 nós cada, e uma camada de saída com 5 nós, cada um deles especificando a probabilidade de determinado dado fazer parte de cada uma das classes (Figura 7.8).

Para o treinamento do sistema de detecção e diagnóstico através de DNN, foram utilizados dois terços dos dados, enquanto o um terço restante foi utilizado para validação. Para o treinamento da rede, foram estabelecidas mil (1000) épocas. Cada época no processo de treinamento da rede neuronal é utilizada para melhorar a matriz de pesos **W**, de acordo com a Equação (5.23). A taxa de aprendizado utilizado foi de 0,01. O treinamento da rede levou 150 minutos.

Figura 7.8 - Estrutura da DNN: Entrada-Camada Oculta-Saída.

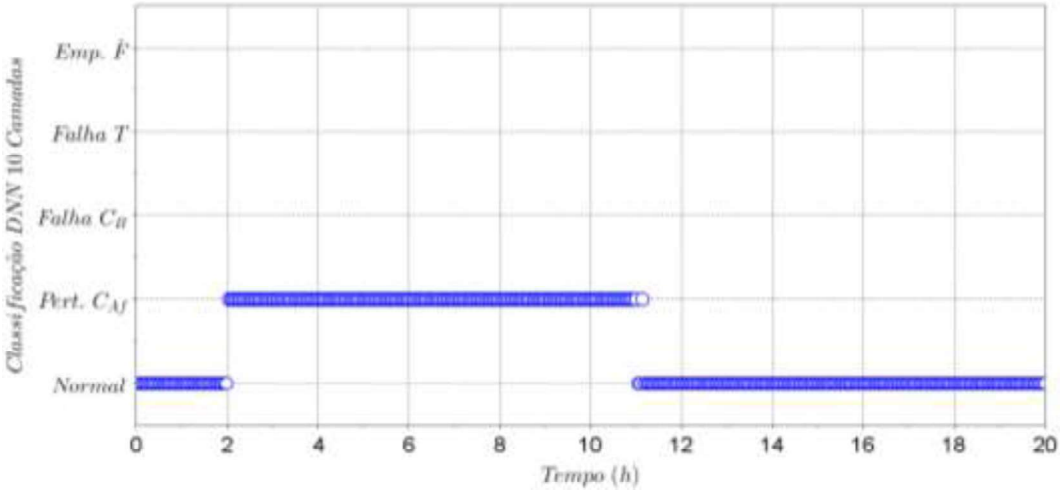


Após o treinamento e validação da rede, foi armazenada a matriz peso **W** e as informações pertinentes à rede DNN, como o número de neurônios de cada camada oculta da rede e a taxa de aprendizado cuja rede foi treinada, que posteriormente foi utilizada para a classificação de novos dados. Para a obtenção da classe dos dados testados foi estabelecida a maior probabilidade de fazer parte de determinada classe para cada instante, e esses dados foram plotados para melhor visualização. Nesse caso, todas as camadas ocultas apresentavam

função de ativação logarítmica sigmoidal, enquanto a camada de saída apresenta função de ativação linear.

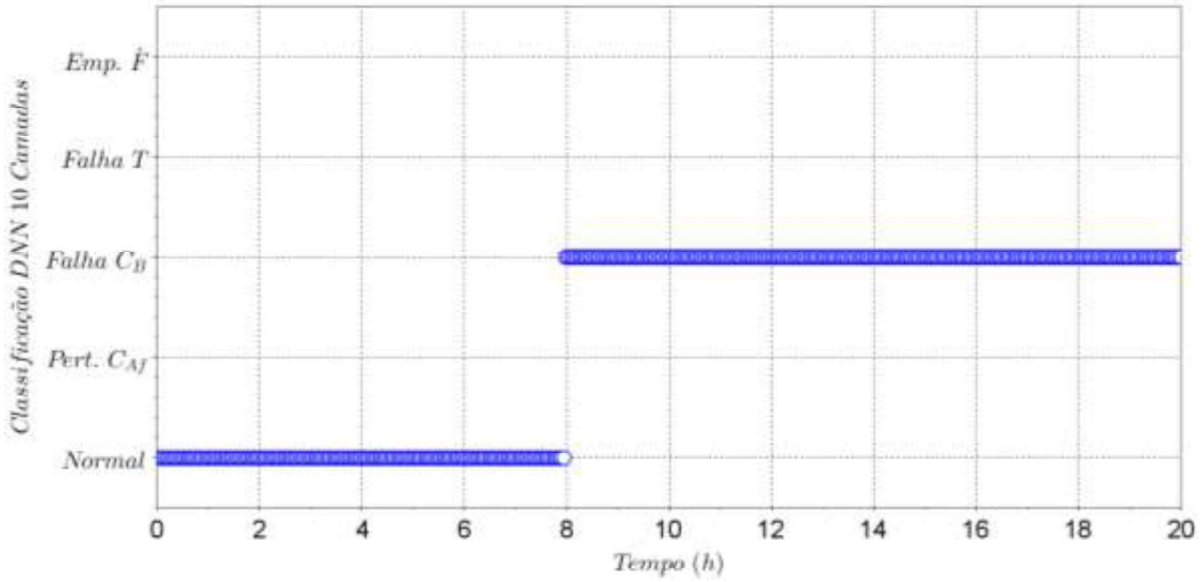
A Figura 7.9 apresenta a classificação dos dados da perturbação em C_{Af}, aplicada no instante de 2 h e removida no instante de 10 h. A perturbação passou a ser detectada após um intervalo de amostragem (0,05 h). Após a remoção da perturbação, o sistema de detecção levou vinte intervalos de amostragem (1,00 h) para detectar novamente a operação normal.

Figura 7.9 - Rótulos obtidos através do processo de detecção da perturbação C_{Af} - DNN.



A Figura 7.10 apresenta a classificação dos dados da falha em C_B, aplicada no instante de 8 h. A falha passou a ser detectada após um intervalo de amostragem (0,05 h).

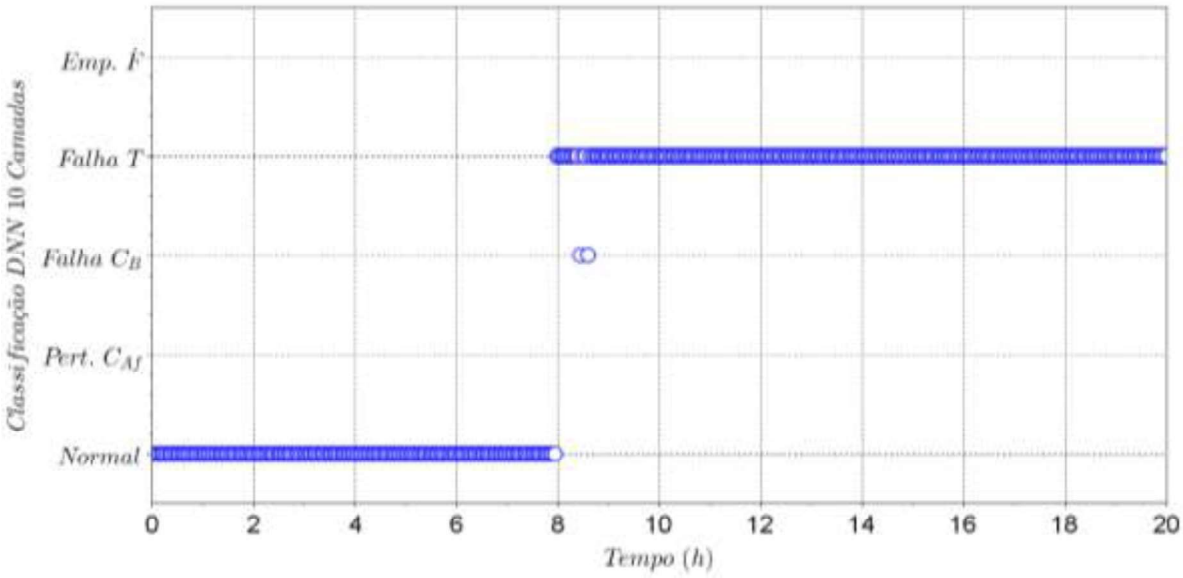
Figura 7.10 - Rótulos obtidos através do processo de detecção da falha C_B - DNN.



A Figura 7.11 apresenta a classificação dos dados da falha no sensor T, aplicada no instante de 8 h. A falha foi detectada após um intervalo de amostragem (0,05 h). Durante dois

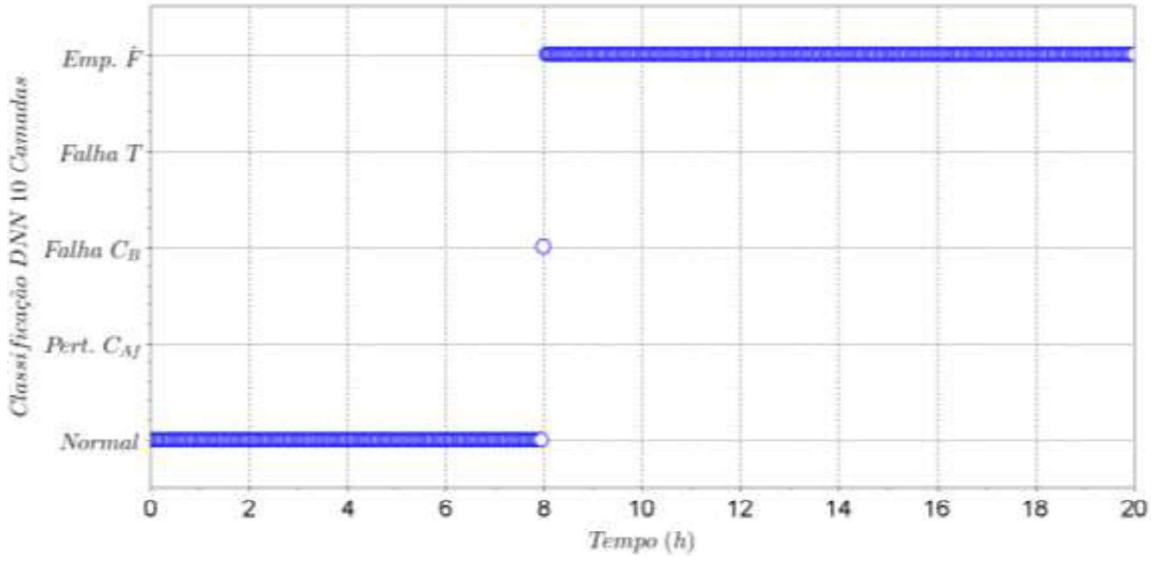
intervalos de amostragem, após alguns instantes que a Falha *T* foi detectada, o sistema de detecção de falhas identificou a Falha C_B ao invés da Falha *T*.

Figura 7.11 - Rótulos obtidos através do processo de detecção da falha *T* - DNN.



A Figura 7.12 apresenta a classificação dos dados do emperramento em *F*, aplicada no instante de 8 h. A falha foi detectada após um intervalo de amostragem (0,05 h). Durante um intervalo de amostragem, antes do sistema de detecção de falhas detectar o emperramento *F*, o FDD detectou erroneamente a falha C_B.

Figura 7.12 - Rótulos obtidos através do processo de detecção do Emperramento *F* - DNN.



Foram contabilizados todos os instantes nos quais o comportamento do sistema era normal e todos os instantes em que o sistema estava passando pelas diferentes falhas treinadas,

de forma que foram criados índices que mostram qual a porcentagem de identificações corretas ou não, de acordo com a Equações (3.1) e (3.2), para todas as operações com falhas testadas para o sistema de detecção e diagnóstico através de DNN.

A Tabela 7.2 apresenta os índices encontrados para o sistema de detecção e diagnóstico através do DNN.

Tabela 7.2 - Índices encontrados - DNN.

Oper.	Índices - DNN				
	DF/F	DF/N	DN/N	DN/F	DF_i/F_j
Pert. C_{Af}	1,00	0,00	1,00	0,00	0,00
Falha C_B	1,00	0,00	1,00	0,00	0,00
Falha T	0,99166	0,00	1,00	0,00	0,00834
Emp. F_f	0,99583	0,00	1,00	0,00	0,00417

O método de detecção e diagnóstico de falhas através de redes neuronais artificiais com múltiplas camadas ocultas (DNN), comparado aos outros métodos de detecção e diagnóstico de falhas resulta em detecção e diagnóstico praticamente instantâneos, levando em todos os cenários testados somente um instante para a detecção (apesar que em dois cenários, Falha C_{Af} e Emperramento F, o DNN apresentar erros de classificação). Porém, o treinamento de uma rede neuronal artificial demanda uma alta capacidade computacional, levando cerca de 150 minutos para o seu treinamento.

7.5.3. Redes Neuronais Artificiais Linguísticas

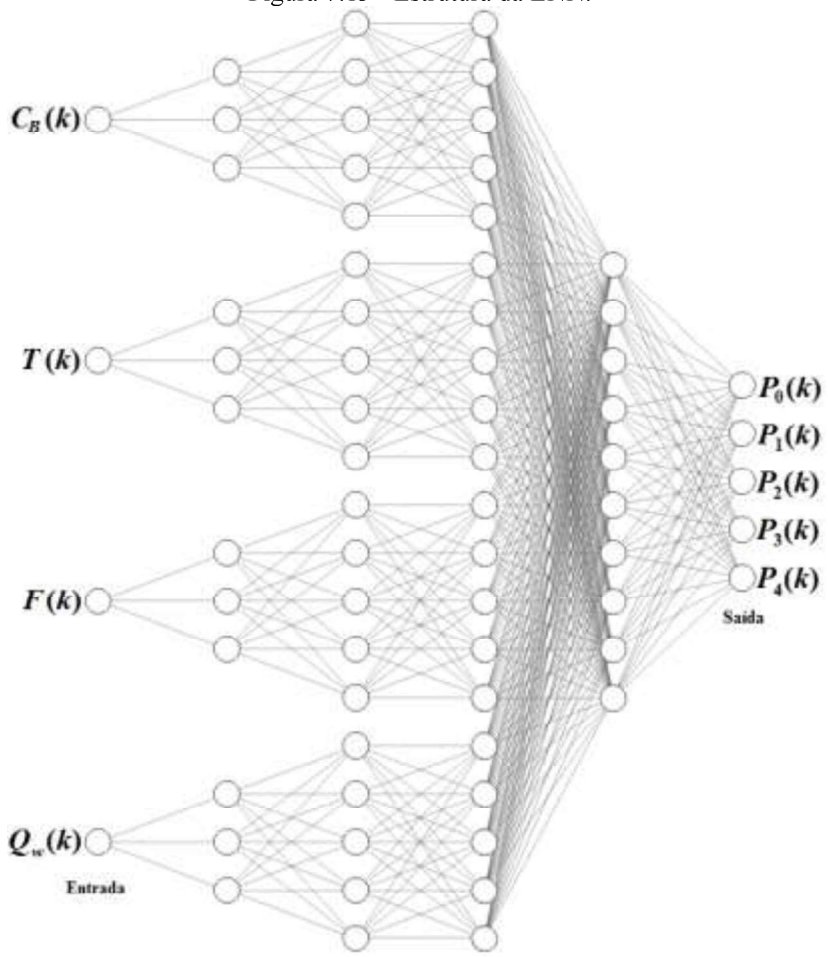
Para as redes neuronais artificiais com variáveis linguísticas, proposta na Seção 7.3 (*Linguistic Neural Networks*, LNN) e algoritmo *Back Propagation*, foi necessário treinar a rede neuronal com as informações de treinamento de todas as falhas com as variáveis controladas, C_B e T, e variáveis manipuladas, $\dot{F}$ e $\dot{Q}_w$, como entradas do modelo LNN. A Figura 7.13 apresenta a rede LNN.

Para a saída do modelo LNN, foram estabelecidas probabilidades para que cada dado faça parte de cada uma das classes, sendo essas classes as operações faltosas (Pert. C_{Af} - 1, Falha C_B - 2, Falha T - 3 e Emperramento F - 4) e a operação normal (0). Para a saída do treinamento e validação, foram estabelecidas juntamente às variáveis de treinamento e validação o vetor de probabilidades $P = [P_0 \quad P_1 \quad P_2 \quad P_3 \quad P_4]^T$, para $P_i = 1$ se i = falha, e $P_j = 0$ se $j \neq$ falha.

A rede LNN é uma rede neuronal especial, cada entrada da rede $x = [C_B \quad T \quad \dot{F} \quad \dot{Q}_w]^T$ passa por uma classificação linguística através de *Fuzzy C-means* sendo que existem 4 saídas para *Fuzzy*, $z = [MB \quad B \quad A \quad MA]^T$, chamadas de variáveis linguísticas. As saídas de cada *Fuzzy C-means* linguística (4 sistemas) são a entrada da rede neuronal, cuja entrada nesse caso

possui 16 neurônios, a camada intermediária possui 10 nós e a camada de saída possui 5 neurônios ($y = [P_0 \quad P_1 \quad P_2 \quad P_3 \quad P_4]^T$).

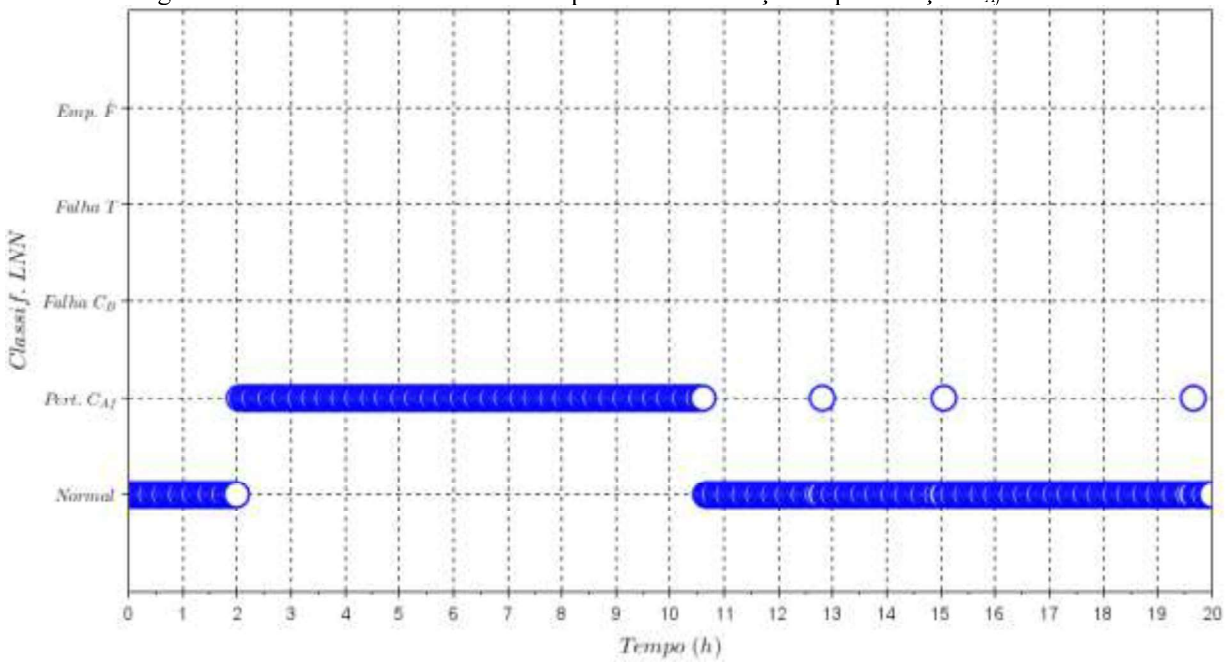
Figura 7.13 - Estrutura da LNN.



Para o treinamento do sistema de detecção e diagnóstico através de LNN, foram utilizados dois terços dos dados, enquanto o terço restante foi utilizado para validação. Para os treinamentos de cada rede das variáveis de entrada, foram estabelecidas mil (1000) épocas de forma empírica. O treinamento da rede LNN levou em torno de 100 min para 1000 épocas. Após o treinamento das redes com variáveis linguísticas, as saídas dessas redes são responsáveis para a entrada da última rede neuronal, com taxa de aprendizado de 0,01 e gradiente decrescente. A função de ativação utilizada em todas as camadas será a logarítmica sigmoidal.

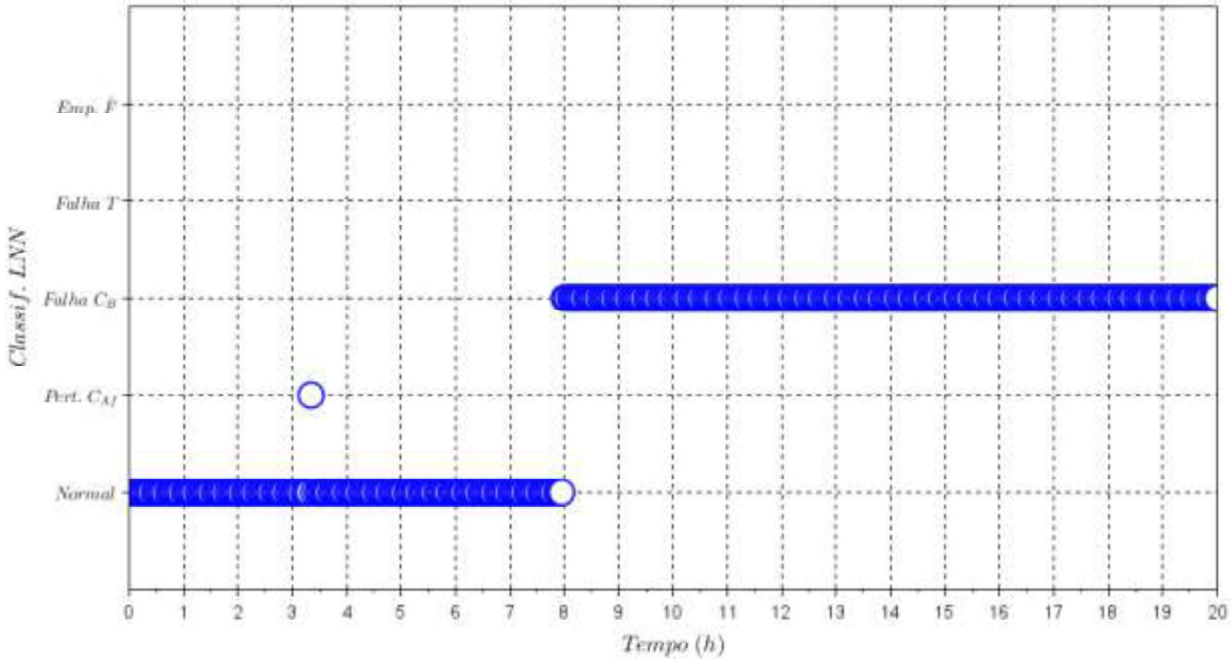
A Figura 7.14 apresenta a classificação dos dados da perturbação em C_{Af}, aplicada no instante de 2 h e removida no instante de 10 h. A perturbação não foi detectada em nenhum instante em que foi aplicada.

Figura 7.14 - Rótulos obtidos através do processo de detecção da perturbação C_{Af} - LNN.



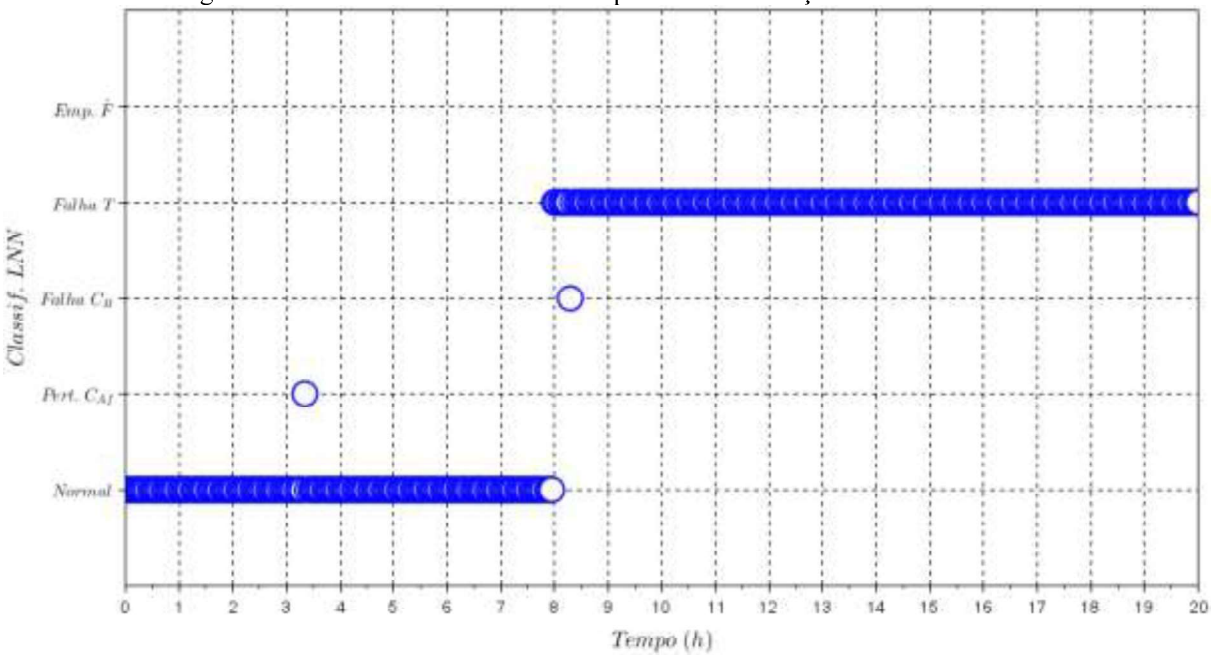
A Figura 7.15 apresenta a classificação dos dados da falha em C_B, aplicada no instante de 8 h. A falha passou a ser detectada após um intervalo de amostragem (0,05 h).

Figura 7.15 - Rótulos obtidos através do processo de detecção da falha C_B - LNN.



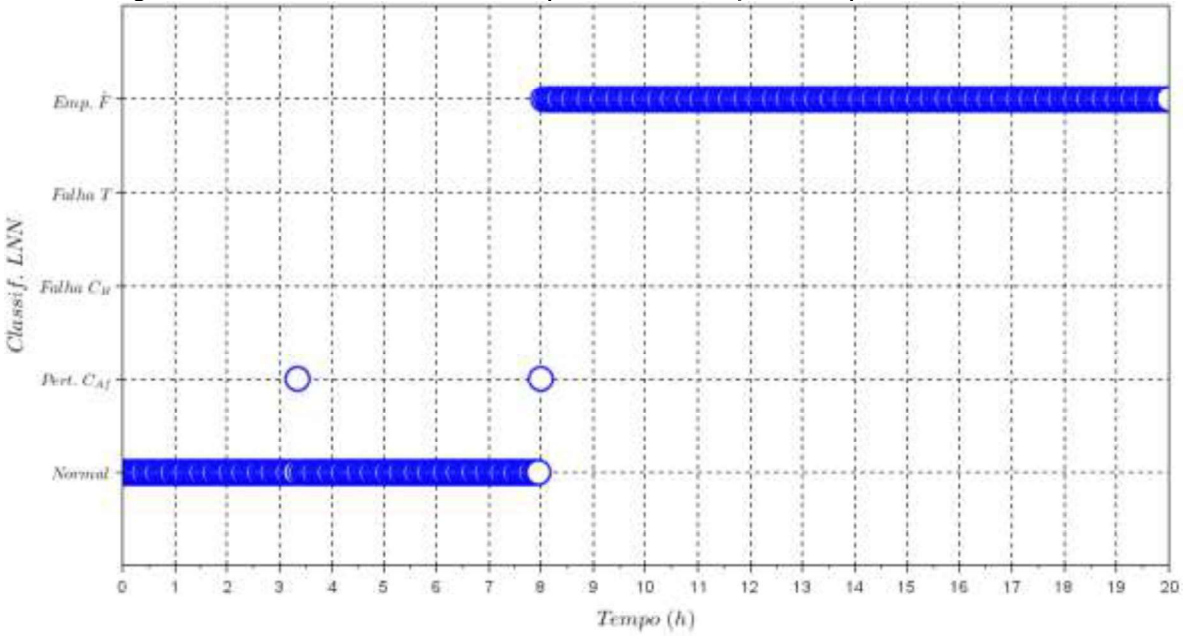
A Figura 7.16 apresenta a classificação dos dados da falha no sensor T, aplicada no instante de 8 h. A falha foi detectada após um intervalo de amostragem (0,05 h).

Figura 7.16 - Rótulos obtidos através do processo de detecção da falha *T* - LNN.



A Figura 7.17 apresenta a classificação dos dados do emperramento em *F*, aplicada no instante de 8 h. A falha foi detectada após um intervalo de amostragem (0,05 h).

Figura 7.17 - Rótulos obtidos através do processo de detecção do Emperramento *F* - LNN.



A Tabela 7.3 apresenta os índices encontrados para o sistema de detecção e diagnóstico através do LNN.

Tabela 7.3 - Índices encontrados - LNN.

Oper.	Índices - LNN				
	DF/F	DF/N	DN/N	DN/F	DF_i/F_j
Pert. C_{Af}	0,9381	0,0067	0,9933	----	0,0619
Falha C_B	1,0000	0,0133	0,9867	0,0000	---
Falha T	0,9953	0,0000	1,0000	----	0,0047
Emp. F_f	1,0000	0,0133	0,9867	0,0000	---

Comparando os três métodos, a Tabela 7.4 apresenta os resultados do SNN, DNN e LNN para o primeiro estudo de caso.

Tabela 7.4 - Índices comparados - SNN, DNN e LNN.

Oper.		Índices				
		DF/F	DF/N	DN/N	DN/F	DF_i/F_j
Pert. C_{Af}	SNN	1,0000	0,0000	1,0000	0,0000	----
	DNN	1,0000	0,0000	1,0000	0,0000	0,0000
	LNN	0,9381	0,0067	0,9933	----	0,0619
Falha C_B	SNN	1,0000	0,0000	1,0000	0,0000	
	DNN	1,0000	0,0000	1,0000	0,0000	0,0000
	LNN	1,0000	0,0133	0,9867	0,0000	----
Falha T	SNN	1,0000	0,0000	1,0000	0,0000	----
	DNN	0,99166	0,0000	1,0000	0,0000	0,00834
	LNN	0,9953	0,0000	1,0000	----	0,0047
Emp. F_f	SNN	1,0000	0,0000	1,0000	0,0000	----
	DNN	0,99583	0,0000	1,0000	0,0000	0,00417
	LNN	1,0000	0,0133	0,9867	0,0000	---

7.6. Estudo de Caso 2 - Planta Química

Para ilustrar a aplicação da detecção e identificação de falhas através de redes neuronais de aprendizagem profunda, foi utilizado também uma planta química composta por dois tanques e um tanque *flash*, explicada anteriormente na Seção 3.6.

7.6.1. Redes Neuronais Artificiais Supervisionadas com Única Camada

Para as redes neuronais artificiais de única camada oculta (*Shallow Neural Networks*, SNN) foi necessário treinar a rede neuronal com as informações de treinamento de todas as variáveis controladas, H_1, H_2, H_3, T_1, T_2 e T_3, e variáveis manipuladas, Ff_1 (F_1), Ff_2 (F_2), F_R (F_3), Q_1, Q_2 e Q_3, como entradas do modelo SNN, para todas as falhas e estados estacionários.

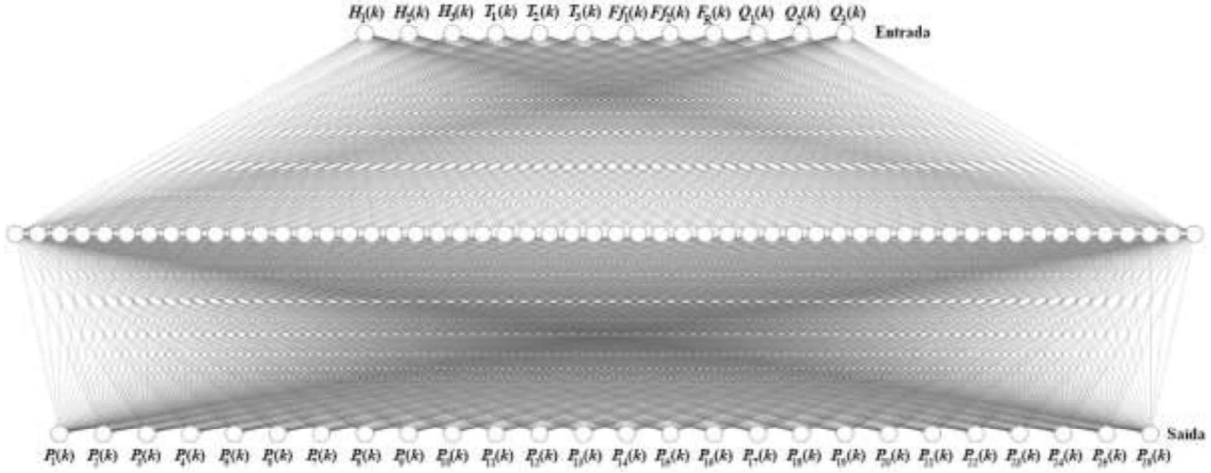
Para a saída do modelo SNN, foram estabelecidas probabilidades para que cada dado faça parte de cada uma das classes, sendo essas classes as duas operações faltosas em cada uma das variáveis controladas ou manipuladas, e os dois estados estacionários 1 e 2, de acordo com a Tabela 7.5. Para a saída do treinamento e validação, foram estabelecidas juntamente às variáveis de treinamento e validação o vetor de probabilidades $P = [P_1 \quad P_2 \quad \dots \quad P_{25} \quad P_{26}]^T$, para $P_i = 1$ se i = falha ou estado estacionário detectado, e $P_j = 0$ se $j \neq$ falha ou estado estacionário detectado.

Tabela 7.5 - Rótulos aplicados a diferentes situações - Estudo de caso 2.

Estado	Amplitude	Rótulo	Estado	Amplitude	Rótulo
Q_3 (ee1)	0,70	1	ee2	-	14
F_R (ee1)	1,30	2	H_1 (ee2)	0,70	15
T_3 (ee1)	0,98	3	T_1 (ee2)	1,02	16
H_3 (ee1)	1,30	4	F_{f1} (ee2)	0,70	17
Q_2 (ee1)	070	5	Q_1 (ee2)	1,30	18
F_{f2} (ee1)	1,30	6	H_2 (ee2)	0,70	19
T_2 (ee1)	0,98	7	T_2 (ee2)	1,02	20
H_2 (ee1)	1,30	8	Ff_2 (ee2)	0,70	21
Q_1 (ee1)	0,70	9	Q_2 (ee2)	1,30	22
F_{f1} (ee1)	1,30	10	H_3 (ee2)	0,70	23
T_1 (ee1)	0,98	11	T_3 (ee2)	1,02	24
H_1 (ee1)	1,30	12	F_R (ee2)	0,70	25
ee1	-	13	Q_3 (ee2)	1,30	26

A rede então foi definida com a entrada $x = [H_i \quad T_i \quad F_i \quad Q_i]^T$, com i = 1 até 3, e saída $y = [P_1 \quad P_2 \quad \dots \quad P_{25} \quad P_{26}]^T$. A estrutura da rede foi definida com 12 nós na camada de entrada, cada nó para cada uma das variáveis do vetor x, com uma camada interna de neurônios, sendo uma única camada oculta com 54 nós, e uma camada de saída com 26 nós, cada um dele especificando a probabilidade de determinado dado fazer parte de cada uma das classes (Figura 7.18).

Figura 7.18 - Estrutura da SNN - Estudo de Caso 2.



Para o treinamento do sistema de detecção e diagnóstico através de SNN, foram utilizados dois terços dos dados, enquanto que o um terço restante foi utilizado para validação. Para o treinamento da rede, foram estabelecidas mil (1000) épocas, de forma a minimizar o MSE entre os dados de validação e os dados de treinamento. Cada época no processo de treinamento da rede neuronal é utilizada para melhorar a matriz de pesos **W**, de acordo com a Equação (5.23). A taxa de aprendizado utilizado foi de 0,9 e o tempo de treinamento foi de 300 minutos (5 horas).

Após o treinamento e validação da rede, foi armazenada a matriz peso **W** e as informações pertinentes à rede SNN, como o número de neurônios da camada oculta da rede e a taxa de aprendizado cuja rede foi treinada, que posteriormente foi utilizada para a classificação de novos dados. Para a obtenção da classe dos dados testados foi estabelecida a maior probabilidade de fazer parte de determinada classe para cada instante, e esses dados foram plotados para melhor visualização.

Foram contabilizados todos os instantes nos quais o comportamento do sistema era normal e todos os instantes em que o sistema estava passando pelas diferentes falhas treinadas, de forma que foram criados índices que mostram qual a porcentagem de identificações corretas ou não, de acordo com as Equação (3.1), para todas as operações com falhas testadas para o sistema de detecção e diagnóstico através de SNN.

A Tabela 7.6 apresenta os índices encontrados para o sistema de detecção e diagnóstico através do SNN.

Tabela 7.6 - Índices encontrados - SNN - Estudo de Caso 2

Índices – SNN									
Oper.	DF/F	DF/N	DN/N	DN/F	Oper.	DF/F	DF/N	DN/N	DN/F
Q_3 (ee1)	1,0000	0,0000	1,0000	0,0000	ee2	1,0000	0,0000	1,0000	0,0000
F_R (ee1)	1,0000	0,0000	1,0000	0,0000	H_1 (ee2)	1,0000	0,0000	1,0000	0,0000
T_3 (ee1)	1,0000	0,0000	1,0000	0,0000	T_1 (ee2)	1,0000	0,0000	1,0000	0,0000
H_3 (ee1)	1,0000	0,0000	1,0000	0,0000	Ff_1 (ee2)	1,0000	0,0000	1,0000	0,0000
Q_2 (ee1)	1,0000	0,0000	1,0000	0,0000	Q_1 (ee2)	1,0000	0,0000	1,0000	0,0000
Ff_2 (ee1)	1,0000	0,0000	1,0000	0,0000	H_2 (ee2)	1,0000	0,0000	1,0000	0,0000
T_2 (ee1)	1,0000	0,0000	1,0000	0,0000	T_2 (ee2)	1,0000	0,0000	1,0000	0,0000
H_2 (ee1)	1,0000	0,0000	1,0000	0,0000	Ff_2 (ee2)	1,0000	0,0000	1,0000	0,0000
Q_1 (ee1)	1,0000	0,0000	1,0000	0,0000	Q_2 (ee2)	1,0000	0,0000	1,0000	0,0000
Ff_1 (ee1)	1,0000	0,0000	1,0000	0,0000	H_3 (ee2)	1,0000	0,0000	1,0000	0,0000
T_1 (ee1)	1,0000	0,0000	1,0000	0,0000	T_3 (ee2)	1,0000	0,0000	1,0000	0,0000
H_1 (ee1)	1,0000	0,0000	1,0000	0,0000	F_R (ee2)	1,0000	0,0000	1,0000	0,0000
ee1	1,0000	0,0000	1,0000	0,0000	Q_3 (ee2)	1,0000	0,0000	1,0000	0,0000

O método de detecção e diagnóstico de falhas através de redes neuronais artificiais com única camada oculta (SNN), comparado aos outros métodos de detecção e diagnóstico de falhas, resulta em detecção e diagnóstico praticamente instantâneos, levando em todos os cenários testados somente um instante para a detecção. Porém, o treinamento de uma rede neuronal artificial, além das dificuldades inerentes a obtenção de sua estrutura otimizada, demanda uma alta capacidade computacional, levando em alguns casos mais de 300 minutos para o seu treinamento. Os resultados obtidos pela detecção de falhas utilizando o modelo SNN está apresentado no Apêndice A da Figura A.142 até a Figura A.167.

7.6.2. Redes Neuronais Artificiais Supervisionadas com Múltiplas Camadas

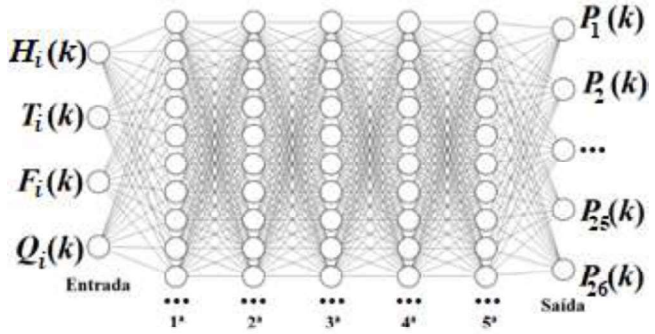
Para as redes neuronais artificiais com múltiplas camadas ocultas (*Deep Neural Networks*, DNN) e algoritmo *Back Propagation*, foi necessário treinar a rede neuronal com as informações de treinamento de todas as variáveis controladas, H_1, H_2, H_3, T_1, T_2 e T_3, e variáveis manipuladas, Ff_1 (F_1), Ff_2 (F_2), F_R (F_3), Q_1, Q_2 e Q_3 como entradas do modelo DNN.

Para a saída do modelo DNN, foram estabelecidas probabilidades para que cada dado faça parte de cada uma das classes, sendo essas classes as operações faltosas e os estados estacionários 1 e 2, considerados as operações normais. Para a saída do treinamento e validação, foram estabelecidas juntamente às variáveis de treinamento e validação o vetor de probabilidades $P = [P_1 \quad P_2 \quad \dots \quad P_{25} \quad P_{26}]^T$, para $P_i = 1$ se i = falha, e $P_j = 0$ se $j \neq$ falha.

A rede então foi definida com a entrada $x = [H_i \quad T_i \quad F_i \quad Q_i]^T$, com i = 1 até 3, e saída $y = [P_1 \quad P_2 \quad \dots \quad P_{25} \quad P_{26}]^T$. A estrutura da rede foi definida com 12 nós na camada de entrada, cada nó para cada uma das variáveis do vetor x, com uma camada interna de neurônios, sendo cinco camadas ocultas com 52 nós cada, e uma camada de saída com 26 nós, cada um deles especificando a probabilidade de determinado dado fazer parte de cada uma das classes (Figura 7.19).

Para o treinamento do sistema de detecção e diagnóstico através de DNN, foram utilizados dois terços dos dados, enquanto que o um terço restante foi utilizado para validação. Para o treinamento da rede, foram estabelecidas mil (1000) épocas. Cada época no processo de treinamento da rede neuronal é utilizada para melhorar a matriz de pesos **W**, de acordo com a Equação (5.23). A taxa de aprendizado utilizado foi de 0,01. O treinamento da rede levou 400 minutos.

Figura 7.19 - Estrutura Simplificada da DNN - Estudo de Caso 2.



Após o treinamento e validação da rede, foi armazenada a matriz peso **W** e as informações pertinentes à rede DNN, como o número de neurônios de cada camada oculta da rede e a taxa de aprendizado cuja rede foi treinada, que posteriormente foi utilizada para a classificação de novos dados. Para a obtenção da classe dos dados testados foi estabelecida a

maior probabilidade de fazer parte de determinada classe para cada instante, e esses dados foram plotados para melhor visualização. Nesse caso, todas as camadas ocultas apresentavam função de ativação logarítmica sigmoidal, enquanto a camada de saída apresenta função de ativação linear.

Foram contabilizados todos os instantes nos quais o comportamento do sistema era normal e todos os instantes em que o sistema estava passando pelas diferentes falhas treinadas, de forma que foram criados índices que mostram qual a porcentagem de identificações corretas ou não, de acordo com a Equações (3.1) e (3.2), para todas as operações com falhas testadas para o sistema de detecção e diagnóstico através de DNN.

A Tabela 7.7 apresenta os índices encontrados para o sistema de detecção e diagnóstico através do DNN.

Tabela 7.7 - Índices encontrados - DNN - Estudo de Caso 2

Índices – DNN									
Oper.	DF/F	DF/N	DN/N	DN/F	Oper.	DF/F	DF/N	DN/N	DN/F
Q_3 (ee1)	1,0000	0,0000	1,0000	0,0000	ee2	0,9900	0,0000	1,0000	0,0100
F_R (ee1)	1,0000	0,0000	1,0000	0,0000	H_1 (ee2)	1,0000	0,0000	1,0000	0,0000
T_3 (ee1)	1,0000	0,0000	1,0000	0,0000	T_1 (ee2)	1,0000	0,0000	1,0000	0,0000
H_3 (ee1)	1,0000	0,0000	1,0000	0,0000	Ff_1 (ee2)	1,0000	0,0000	1,0000	0,0000
Q_2 (ee1)	1,0000	0,0000	1,0000	0,0000	Q_1 (ee2)	1,0000	0,0000	1,0000	0,0000
Ff_2 (ee1)	1,0000	0,0000	1,0000	0,0000	H_2 (ee2)	1,0000	0,0000	1,0000	0,0000
T_2 (ee1)	1,0000	0,0000	1,0000	0,0000	T_2 (ee2)	1,0000	0,0000	1,0000	0,0000
H_2 (ee1)	1,0000	0,0000	1,0000	0,0000	Ff_2 (ee2)	1,0000	0,0000	1,0000	0,0000
Q_1 (ee1)	1,0000	0,0000	1,0000	0,0000	Q_2 (ee2)	1,0000	0,0000	1,0000	0,0000
Ff_1 (ee1)	1,0000	0,0000	1,0000	0,0000	H_3 (ee2)	1,0000	0,0000	1,0000	0,0000
T_1 (ee1)	1,0000	0,0000	1,0000	0,0000	T_3 (ee2)	1,0000	0,0000	1,0000	0,0000
H_1 (ee1)	1,0000	0,0000	1,0000	0,0000	F_R (ee2)	1,0000	0,0000	1,0000	0,0000
ee1	0,9900	0,0000	1,0000	0,0100	Q_3 (ee2)	1,0000	0,0000	1,0000	0,0000

O método de detecção e diagnóstico de falhas através de redes neuronais artificiais com única camada oculta (SNN), comparado aos outros métodos de detecção e diagnóstico de falhas resulta em detecção e diagnóstico praticamente instantâneos, levando em todos os cenários testados somente um instante para a detecção (apesar que em dois cenários, Falha C_{Af} e Emperramento F, o DNN apresentar erros de classificação). Porém, o treinamento de uma rede neuronal artificial demanda uma alta capacidade computacional, levando cerca de 400 minutos para o seu treinamento. Os resultados obtidos pela detecção de falhas utilizando o modelo DNN está apresentado no Apêndice A da Figura A.168 até a Figura A.193.

7.6.3. Redes Neuronais Artificiais Linguísticas

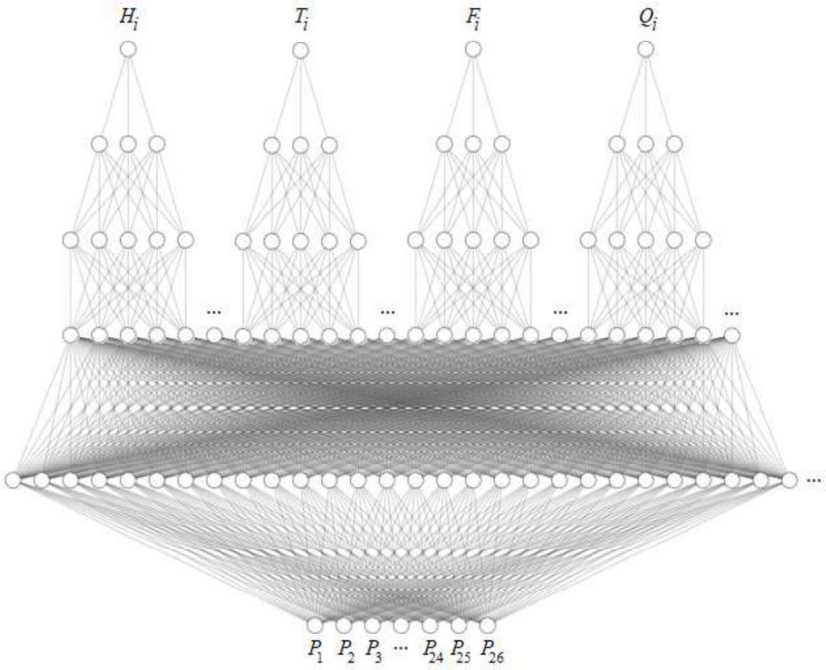
Para as redes neuronais artificiais com variáveis linguísticas, proposta na Seção 7.3 (*Linguistic Neural Networks*, LNN) e algoritmo *Back Propagation*, foi necessário treinar a rede

neuronal com as informações de treinamento de todas as falhas com as variáveis controladas, H_1, H_2, H_3, T_1, T_2 e T_3 e variáveis manipuladas F_1 (Ff_1), F_2 (Ff_2), F_3 (F_R), Q_1, Q_2 e Q_3 como entradas do modelo LNN. A Figura 7.20 apresenta a rede LNN para o estudo de caso 2.

Para a saída do modelo LNN, foram estabelecidas probabilidades para que cada dado faça parte de cada uma das classes, sendo essas classes as operações faltosas e os estados estacionários de acordo com a Tabela 3.5. Para a saída do treinamento e validação, foram estabelecidas juntamente às variáveis de treinamento e validação o vetor de probabilidades $P = [P_1 \quad P_2 \quad \dots \quad P_{25} \quad P_{26}]^T$, para $P_i = 1$ se i = falha, e $P_j = 0$ se $j \neq$ falha.

A rede LNN é uma rede neuronal especial, cada entrada da rede $x = [H_i \quad T_i \quad F_i \quad Q_i]^T$ para i = 1, 2, 3, passa por uma classificação através de *Fuzzy C-means*, cuja saídas $z = [MB \quad B \quad A \quad MA]^T$ são chamadas de variáveis linguísticas. As saídas de cada *Fuzzy C-means* linguística (12 saídas) são as entradas da rede neuronal, cuja entrada nesse caso possui 48 neurônios, a camada intermediária possui 52 nós e a camada de saída possui 26 neurônios ($y = [P_1 \quad P_2 \quad \dots \quad P_{25} \quad P_{26}]^T$).

Figura 7.20 - Estrutura Simplificada da LNN - Estudo de Caso 2.



Para o treinamento do sistema de detecção e diagnóstico através de LNN foram utilizados dois terços dos dados, enquanto o terço restante foi utilizado para validação. Para os treinamentos de cada rede das variáveis de entrada, foram utilizados algoritmos de *fuzzy c-means* para 4 núcleos, designados MB (Muito Baixo), B (Baixo), A (Alto) e MA (Muito Alto). Após o treinamento das redes, as saídas dessas redes são responsáveis para a entrada da última rede, que também passará por mil épocas para o treinamento, com taxa de aprendizado de 0,1

e gradiente decrescente. A função de ativação utilizada em todas as camadas será a logarítmica sigmoidal.

A Tabela 7.8 apresenta os índices encontrados para o sistema de detecção e diagnóstico através do DNN.

Tabela 7.8 - Índices encontrados - LNN - Estudo de Caso 2

Índices - LNN									
Oper.	DF/F	DF/N	DN/N	DN/F	Oper.	DF/F	DF/N	DN/N	DN/F
Q_3 (ee1)	1,0000	0,0000	1,0000	0,0000	ee2	1,0000	0,0000	1,0000	0,0000
F_R (ee1)	1,0000	0,0000	1,0000	0,0000	H_1 (ee2)	1,0000	0,0000	1,0000	0,0000
T_3 (ee1)	1,0000	0,0000	1,0000	0,0000	T_1 (ee2)	1,0000	0,0000	1,0000	0,0000
H_3 (ee1)	1,0000	0,0000	1,0000	0,0000	Ff_1 (ee2)	1,0000	0,0000	1,0000	0,0000
Q_2 (ee1)	1,0000	0,0000	1,0000	0,0000	Q_1 (ee2)	1,0000	0,0000	1,0000	0,0000
Ff_2 (ee1)	1,0000	0,0000	1,0000	0,0000	H_2 (ee2)	1,0000	0,0000	1,0000	0,0000
T_2 (ee1)	1,0000	0,0000	1,0000	0,0000	T_2 (ee2)	1,0000	0,0000	1,0000	0,0000
H_2 (ee1)	1,0000	0,0000	1,0000	0,0000	Ff_2 (ee2)	1,0000	0,0000	1,0000	0,0000
Q_1 (ee1)	1,0000	0,0000	1,0000	0,0000	Q_2 (ee2)	1,0000	0,0000	1,0000	0,0000
Ff_1 (ee1)	1,0000	0,0000	1,0000	0,0000	H_3 (ee2)	1,0000	0,0000	1,0000	0,0000
T_1 (ee1)	1,0000	0,0000	1,0000	0,0000	T_3 (ee2)	1,0000	0,0000	1,0000	0,0000
H_1 (ee1)	1,0000	0,0000	1,0000	0,0000	F_R (ee2)	1,0000	0,0000	1,0000	0,0000
ee1	1,0000	0,0100	0,9900	0,0000	Q_3 (ee2)	1,0000	0,0000	1,0000	0,0000

Os resultados obtidos pela detecção de falhas utilizando o modelo LNN está apresentado no Apêndice A da Figura A.194 até a Figura A.219.

7.7.Comentários

Os resultados obtidos com métodos de detecção de falhas com aprendizado profundo foram muito próximos dos resultados obtidos com redes neuronais, máquinas de vetores de suporte e lógica difusa. Assim, a metodologia de detecção de falhas através de aprendizado profundo apresenta a capacidade de realizar o propósito para que foi desenvolvida, porém com vantagens quanto a preparação dos dados de entrada, tornando o método útil para sistemas cujos dados não são completamente conhecidos e rotulados.

CAPÍTULO 8
CONCLUSÕES

8.1.Conclusões

Os métodos de detecção de falhas com aprendizado de máquina são técnicas altamente eficazes que podem ser empregadas com uma grande generalização, não importando exclusivamente a estrutura do processo que está sendo analisado, mas sim dados disponíveis de variáveis em sistemas de controle de processos.

Métodos como SVM, ANN e Lógica *Fuzzy*, que utilizam dados históricos do processo, podem ser aplicados como métodos de detecção de falhas em múltiplos tipos de processos. Porém, a necessidade de processar os dados é considerável para esses métodos, existindo a necessidade da normalização e classificação prévia dos dados de treinamento, rotulando esses dados de acordo com as falhas previamente separadas para treinamento. A inclusão de novos dados de treinamento rotulados é difícil para métodos como o SVM e as redes neuronais, sendo necessário um novo treinamento.

Os métodos de detecção de falhas com utilização do SVM possuem vantagens como a velocidade de treinamento, o ajuste do modelo perante os dados e a alta capacidade de classificação de dados através de hiperplanos. Quanto a desvantagens, o SVM possui algumas limitações quanto a inclusão de novos padrões de falha aos dados de treinamento, sendo necessário a rotulagem e posterior treino do sistema de detecção.

As vantagens dos métodos FDD que utilizam redes neuronais artificiais estão principalmente na rapidez de detecção, na possibilidade de detecção *online* e na opção de utilizar treinamento supervisionado ou não supervisionado. Os métodos não supervisionados possuem a capacidade de encontrar as características dos dados treinados, algo que os métodos supervisionados devem ser preparados antes do treinamento. A principal desvantagem dos métodos de detecção de falhas com redes neuronais se encontra no tempo maior de treinamento que os outros métodos estudados.

O método de detecção de falhas com lógica *fuzzy* possui vantagens quando não existir a necessidade de treinar o modelo, baseado em funções de pertinência e alta adaptabilidade para diferentes sistemas químicos. Quanto a desvantagem, o FDD com lógica *fuzzy* necessita da normalização de todos os dados e do conhecimento empírico das características dos dados a serem classificados.

Métodos com aprendizado profundo tornam-se necessários e de extrema importância em detecção de falhas, justamente pela presença de muitos padrões não conhecidos ou não

identificados pelo engenheiro ou responsável pelo processo. Os métodos de detecção de falhas com aprendizado profundo possuem a capacidade de através de uma grande base de dados, identificar padrões e classificá-los a partir de propriedades extraídas automaticamente, sem a necessidade de preparos maiores dos dados de treinamento.

Resultados obtidos no presente trabalho mostram que métodos de aprendizado profundo e os métodos híbridos podem economizar uma quantidade importante de tempo e esforço no preparo dos dados, apresentando resultados compatíveis com outros métodos que utilizam aprendizado supervisionado, como o SVM e a ANN.

O método de detecção de falhas LNN apresenta vantagens claras perante as outras metodologias, uma vez que necessitou de menos etapas de preparo dos dados, utilizou algoritmos inteligentes capazes de identificar as características dos dados de entrada, informando poucos parâmetros para sua efetiva utilização, e fez o uso de algoritmo de classificação do algoritmo *fuzzy c-means*. Os resultados do LNN foram compatíveis com os outros métodos estudados neste trabalho. A desvantagem do método LNN perante outras metodologias como o SVM foi somente o tempo de treinamento, compatível com o tempo gasto com outros métodos que utilizam redes neuronais.

8.2.Sugestões para Trabalhos Futuros

Como sugestões para trabalhos futuros, pode-se citar:

- Investigação de detecção de falhas em processos mais complexos;
- Utilização para detecção de falhas em dados reais de plantas químicas;
- Investigação de falhas simultâneas em processos químicos;
- Hibridismo entre lógica *fuzzy* e máquinas de vetores de suporte;
- Hibridismo entre máquinas de vetores de suporte e redes neuronais artificiais;
- Desenvolvimento de pacote de detecção de falhas para *softwares* matemáticos populares;

REFERÊNCIAS BIBLIOGRÁFICAS

ABDULGHAFOUR, M.; EL-GAMAL, M.A. A Fuzzy Logic System for Analog Fault Diagnosis, Proc. ISCAS '96, Connecting the World, 1996 IEEE International Symposium on Circuits and Systems, Vol. 1, pp. 97-100, 1996.

ADMUTHE, L.; APTE, S.; AMUTHE, S. Topology and parameter optimization of ANN using genetic algorithm for application of textiles. In: IEEE INTERNATIONAL WORKSHOP ON INTELLIGENT DATA ACQUISITION AND ADVANCED COMPUTING SYSTEMS: TECHNOLOGY AND APPLICATIONS (IDAACS 2009). Rende (Cosenza), Itália, 2009. https://doi.org/10.1109/IDAACS.2009.5342981

AGGARWAL, C.C.; HAN, J.; WANG, J.; YU, P.S. A Framework for Clustering Evolving Data Streams, In: Proc. 29th Int. Conf. Very Large Data Bases, p. 81-92, 2003. https://doi.org/10.1016/B978-012722442-8/50016-1

AGHBASHLO, M.; KIANMEHR, M.H.; NAZGHELICHI, T.; RAFIEE, S. Optimization of an Artificial Neural Network Topology for Predicting Drying Kinetics of Carrot Cubes Using Combined Response Surface and Genetic Algorithm. Drying Technology, v. 29 (7), 2011. p. 770-779. https://doi.org/10.1080/07373937.2010.538819

AGUIRRE, L. A. Introdução à Identificação de Sistemas - Técnicas Lineares e Não-Lineares Aplicadas a Sistemas Reais, 3ª ed., UFMG, 2007.

AHALT, S.C.; FOWLER, J.E. Vector Quantization Using Artificial Neural Network Models. Adaptive Methods and Emerging Techniques for Signal Processing and Communications, p. 42-61, 1993.

AHALT, S.C.; KRISHNAMURTY, A.K.; CHEN, P.; MELTON, D.E. Competitive Learning Algorithms for Vector Quantization. Neural Networks, v. 3, n. 3, p. 277-290, 1990. https://doi.org/10.1016/0893-6080(90)90071-R

AIZERMAN, M.A., BRAVERMAN E.M., ROZONOÉR, L.I. Theoretical foundations of the potential function method in pattern recognition learning. Automation and Remote Control, 25:821-837, 1964.

AMARAL, J.F.M.; TANSCHEIT, R.; PACHECO, M.A.; VELASCO, M.M.; FEITOSA, R.; SCHIFINI, R. Análise não Destrutiva de Estruturas de Aço usando a Técnica do Campo Magnético de Fuga e Redes Neurais, Anais VI SBAI, pp. 331-336, Bauru, 2003.

ANDERS, U.; KORN, O. Model selection in neural networks. Neural Networks, v. 12 (2), 1999. p. 309-323. https://doi.org/10.1016/S0893-6080(98)00117-8

ANGELI, C.; CHATZINIKOLAOU, A. On-line Fault Detection Techniques for Technical Systems: A survey, International Journal of Computer Science and Applications, v. I (1), 2004. p. 12-30.

ARCOVERDE, G.F.B.; ALMEIDA, C.M.; XIMENES, A.C.; MAEDA, E.E.; ARAÚJO, L.S. Identificação de Áreas Prioritárias Para Recuperação Florestal com o Uso de Rede Neural de Mapas Auto-Organizáveis. Boletim de Ciências Geodésicas, v. 17, n. 3, p. 379-400, 2011.
https://doi.org/10.1590/S1982-21702011000300004

ARIFOVIC, J.; GENÇAY, R. Using genetic algorithms to select architecture of a feedforward artificial neural network. Physica A: Statistical Mechanics and its Applications, v. 289 (3-4), 2001. p. 574-594.
https://doi.org/10.1016/S0378-4371(00)00479-9

ARLOT, S. A survey of cross-validation procedures for model selection. Statistics Surveys, v. 4, 2010. p. 40-79.
https://doi.org/10.1214/09-SS054

ARMENGOL, J.; VEHF, J.; SAINZ, M. À.; HERRERO, P. Fault Detection in a Pilot Plant Using Interval Models and Multiple Sliding Time Windows, In: SAFEPROCESS, 2003.
https://doi.org/10.1016/S1474-6670(17)36571-0

ARONSZAJN, N. Theory of reproducing kernels. Trans. Amer. Math. Soc., 686:337-404, 1950.
https://doi.org/10.21236/ADA296533

ASSIDJO, E.; YAO, B.; AMANE, D.; ADO, G.; AZZARO-PANTEL, C.; DAVIN, A. Industrial Brewery Modelling by Using Neural Network. Journal of Applied Sciences, v. 6 (8), 2006. p. 1858-1862.
https://doi.org/10.3923/jas.2006.1858.1862

ATTIK, M.; BOUGRAIN, L.; ALEXANDRE, F. Optimal brain surgeon variants for feature selection, In: INTERNATIONAL JOINT CONFERENCE ON NEURAL NETWORKS (IJCNN), 2004.

BA, L.J.; CARUANA, R. Do Deep Nets Really Need to be Deep?, Draft for NIPS, 2014.

BAKIOTIS, C.; RAYMOND, J.; RAULT, A. Parameter and Discriminiant Analysis for Jet Engine Mechanical State Diagnosis, In Proc. of The 1979 IEEE Conf. on Decision & Control, Fort Lauderdale, USA, 1979.
https://doi.org/10.1109/CDC.1979.270264

BAKRI, A.E.; BOUMHIDI, I. Fuzzy Model-Based Faults Diagnosis of the Wind Turbine Benchmark, Procedia Computer Science, vol. 127, pp. 464-470, 2018.
https://doi.org/10.1016/j.procs.2018.01.144

BALAKIN, K.V.; COOPER, M.C.; JACOB, V.S.; LEWIS, P.A. Comparative Performance of The FSCL Neural Net and K-Means Algorithm for Market Segmentation. European Journal of Operational Research, v. 93, n. 2, p. 346-357, 1996. https://doi.org/10.1016/0377-2217(96)00046-X

BALLESTEROS-MONCADA, H.; HERRERA-LÓPEZ, E.J.; ANZUREZ-MARÍN, J. Fuzzy Model-Based Observers for Fault Detection in CSTR, ISA Transactions, vol. 59, p. 325-333, 2015. https://doi.org/10.1016/j.isatra.2015.10.006

BATISTA, B.C.F. Soluções de Equações Diferenciais Usando Redes Neurais de Múltiplas Camadas com os Métodos da Descida Mais Íngreme e Levenberg-Marquardt, Dissertação de Mestrado, UFPA, Belém-PA, 2012.

BEARD, R. Failure Accommodation in Linear Systems Through Self-reorganization. Technical Report MVT-71-1, Man Vehicle Laboratory, Cambridge, Mass., 1971.

BELLMAN, R. E. Adaptive Control Processes. Princeton University Press, Princeton, NJ, 1961.

BEZDEK, J. C. Pattern Recognition with Fuzzy Objective Function Algorithms. ISBN: 0-306-40671-3, 1981. https://doi.org/10.1007/978-1-4757-0450-1

BEZERRA, L.G.S.; MARTINS, D.L.; DÓRIA NETO, A.D.; MELO, J.D. de. Detecção e Diagnóstico de Falhas em Sensores de Uma Rede Industrial Foundation Fieldbus Utilizando Sistemas Inteligentes, Anais XI SBAI, Fortaleza, 2013.

BIEROZA, M.; BAKER, A.; BRIDGEMAN, J. Exploratory analysis of excitation-emission matrix fluorescence spectra with self-organizing maps - A tutorial, Education for Chemical Engineers, 2012. https://doi.org/10.1016/j.ece.2011.10.002

BILLINGS, S. A. Identification of Nonlinear Systems - A survey, IEE Procedings-D Control Theory and Applications, v. 127 (6), p. 272-285, 1980. https://doi.org/10.1049/ip-d.1980.0047

BLANKE, M.; IZADI-ZAMENABADI, R.; BOGH, S.; LUNAN, C. Fault-tolerant Control Systems. Control Engineering Practice - CEP, 5(5):693-702, 1997. https://doi.org/10.1016/S0967-0661(97)00051-8

BLANKE, M.; KINNAERT, M.; LUNZE, J.; STAROSWIECKI, M. Diagnosis and Fault-Tolerant Control. Springer Verlag KG W. 2003 https://doi.org/10.1007/978-3-662-05344-7

BLANZ, V.; SCHÖLKOPF, B.; BÜLTHOFF, H.; BURGES, C.; VAPNIK, V.; VETTER, T. Comparison of view-based object recognition algorithms using realistic 3D models. In: VON DER MALSBURG, C.; VON SEELEN, W.; VORBRÜGGEN, J. C.;

SENDHOFF, B. (Eds.), Artificial Neural Networks - ICANN'96, p. 251-256, Berlim, 1996. Springer Lecture Notes in Computer Science, Vol. 1112. https://doi.org/10.1007/3-540-61510-5_45

BOSE, N.K.; LIANG, P. Neural network fundamentals with graphs, algorithms, and applications.Mcgraw-Hill, Inc. EUA, 1996.

BRAGA, A.P.; CARVALHO, A.C.P.L.F.; LUDEMIR, T.B. Redes Neurais Artificiais: Teoria e Aplicações. Rio de Janeiro, Brasil: LTC - Livros Técnicos e Científicos Editora S.A. 2000.

BREIMAN, L.; FRIEDMAN, J.H.; OLSHEN, R.A.; STONE, C.J. Classification and regression trees. Wadsworth Statistics/Probability Series. Wadsworth Advanced Books and Software, Belmont, CA. 1984.

BRUNET, J.; JAUME, D.; LABARRERE, M.; RAULT, A.; VERGÉ, M. Détection et diagnostic de pannes, approche per modélisation, Hermes, Paris, 1990.

BUDURA, G.; BOTOCA, C.; MICLĂU, N. Competitive Learning Algorithms for Data Clustering. The Scientific Journal Facta Universitatis (NIŠ), Series: Electronics and Energetics, v. 19, n. 2, p. 261-269, 2006. https://doi.org/10.2298/FUEE0602261B

BURGES, C.J.C. A tutorial on support vector machines for pattern recognition. Data Mining and Knowledge Discovery, 2, p. 121-167, 1998. https://doi.org/10.1023/A:1009715923555

BURMAN, P. A comparative study of ordinary cross-validation, v-fold cross-validation and the repeated learning-testing methods. Biometrika, 76 (3), 1989. p. 503-514. https://doi.org/10.1093/biomet/76.3.503

CARPENTER, G.A.; GROSSBERG, S. A Massively Parallel Architecture for a Self-Organizing Neural Pattern Recognition Machine, Computer Vision, Graphics, and Image Processing, v. 37, 1, p. 54-115, 1987. https://doi.org/10.1016/S0734-189X(87)80014-2

CASTILLO, I.; EDGAR, T.F. Model Based Fault Detection and Diagnosis, TWCCC Conference, Austin, Texas, 2008.

CATANACH, T. A.; BECK, J. L. Bayesian System Identification Using Auxiliary Stochastic Dynamical Systems, International Journal of Non-Linear Mechanics, v. 94, 2017. p. 72-83. https://doi.org/10.1016/j.ijnonlinmec.2017.03.012

CHAFI, M.S., AKBARZADEH-T, M.-R. MOAVENIAN M. Fault Detection and Isolation in Nonlinear dynamic Systems: A Fuzzy-Neural Approach, IEEE International Fuzzy Systems Conference, 2001.

CHEN, J.; PATTON, R. Robust Model-based Fault Diagnosis for Dynamic Systems. Kluwer, Boston, 1999. https://doi.org/10.1007/978-1-4615-5149-2

CHEN, L.; ZHONG, M.; ZHANG, M. H∞ fault detection for linear singular systems with time-varying delay. International Journal of Control, Automation and Systems, vol. 9, n. 1, p. 9-14, 2011. https://doi.org/10.1007/s12555-011-0102-x

CHIANG, L.; RUSSEL, E.; BRAATZ, R. Fault Detection and Diagnosis in Industrial Systems, Springer, 2001. https://doi.org/10.1007/978-1-4471-0347-9

CHIBANI, A.; CHADLI, M.; DING, S.X.; BRAIEK, N.B. Design of Robust Fuzzy Fault Detection Filter for Polynomial Fuzzy Systems with New Finite Frequency Specifications, Automatica, vol. 93, pp. 42-54, 2018. https://doi.org/10.1016/j.automatica.2018.03.024

CHIU, S. Method and software for extracting fuzzy classification rules by subtractive clustering. In: Fuzzy Information Processing Society, 1996. NAFIPS, 1996 Biennial Conference of the North American, p. 461-465. IEEE. 1996.

CHO, S.B.; KIM, J.H. Rapid Backpropagation Learning Algorithms. Circuits Systems Signal Process, v. 12 (2), 1993. p. 155-175. https://doi.org/10.1007/BF01189872

CLARK, R. A Simplified Instrument Detection Scheme. IEEE Trans. Aerospace Electron. Systems, 14(3): 558-563, 1990. https://doi.org/10.1109/TAES.1978.308680

CLAUMANN, C.A. Desenvolvimento e aplicações de redes neurais wavelets e da teoria de regularização na modelagem de processos. PhD Thesis. Universidade de Santa Catarina, Florianópolis, Brasil. 2003.

CONDORENA, E.G.B. Identificação de processos e controle preditivo com modelo utilizando técnicas de inteligência artificial aplicadas à produção de bioetanol. Tese de doutorado, UNICAMP, 2012.

CONFORTH, M.; MENG, Y. IEEE Swarm Intelligence Symposium, St. Louis, MO, EUA. 2008.

CORTES, C.; VAPNIK, V. Support vector networks. Machine Learning, 20:273-297, 1995. https://doi.org/10.1007/BF00994018

COSTA, J.A.F.; GONÇALVES, M.L.; NETTO, M.L.A. Visualização e Análise de Agrupamentos Usando Redes Auto Organizáveis, Segmentação de Imagens e Índices de Validação. Learning and Nonlinear Models (L&NLM) - Journal of The Brazilian

Neural Network Society, v. 9, n. 2, p. 91-103, 2011.
https://doi.org/10.21528/LNLM-vol9-no2-art2

CYBENKO, G. Aproximation by Superpositions of a Sigmoidal Function. Mathematics of Control, Signals and Systems, v. 2, 1989. p. 303-314.
https://doi.org/10.1007/BF02551274

DA SILVA, I.N.; SPATTI, D.H.; FLAUZINO, R.A. Redes Neurais Artificiais Para Engenharia e Ciências Aplicadas - Curso Prático. São Paulo: Artliber Editora, p. 399, 2010.

DALTON, T. Intensive Course Modern Aspects of Control Engineering and Low Cost Automation. Cluj-Napoca University, 16p. 1999.

DEHESTANI, D.; EFTEKHARI, F.; GUO, Y.; LING, S.H.; SU, S.; NGUYEN, H. Online Support Vector Machine Application for Model Based Fault Detection and Isolation of HVAC System. International Journal of Machine Learning and Computing. 1, 2011. p. 66-72.
https://doi.org/10.7763/IJMLC.2011.V1.10

DENG, L.; YU, D. Deep Learning: Methods and Aplications. Foundations and Trends in Signal Processing, v. 7, n. 3-4, p. 197-387, 2013.
https://doi.org/10.1561/2000000039

DEVILLE, Y.; LAU, K.K. Logic program synthesis. Journal of Logic Programming, 19(20), 321-350, 1994.
https://doi.org/10.1016/0743-1066(94)90029-9

DHIMISH, M.; HOLMES, V.; MEHRDADI, B.; DALES, M. Comparing Mamdani Sugeno Fuzzy Logic and RBF ANN Network for PV Fault Detection, Renewable Energy, v. 117, 2018. p. 257-274.
https://doi.org/10.1016/j.renene.2017.10.066

DIERCKX, P. Curve and Surface Fitting with Splines. Monographs on Numerical Analysis. Clarendon Press, Oxford, 1993.

DING, S.X. Model-Based Fault Diagnosis Techniques: Design Schemes, Algorithms and Tools. Springer, 2013.
https://doi.org/10.1007/978-1-4471-4799-2

DING, S.X.; SCHNEIDER, S.; DING, E.L.; REHM, A.; GMBH, R.B. Advanced Model-Based Diagnosis of Sensor Faults in Vehicle Dynamics Control Systems. In: 16th IFAC WORLD CONGRESS, 2005.
https://doi.org/10.3182/20050703-6-CZ-1902.01887

DONGES, N. Recurrent Neural Networks and LSTM. Disponível em: < https://towardsdatascience.com/recurrent-neural-networks-and-lstm-4b601dd822a5> Acesso em: 15 de julho de 2018.

DUNN, J.C. A Fuzzy Relative of the ISODATA Process and Its Use in Detecting Compact Well-Separated Clusters". Journal of Cybernetics. 3 (3): 32-57, 1973. https://doi.org/10.1080/01969727308546046

ELMAN, J.L. Finding structure in time. Cognitive Science, 14(2), 179-211, 1990. https://doi.org/10.1207/s15516709cog1402_1

FALHMAN, S.E.; LEBIERE, C. The cascade-correlation learning architecture. In: TOURETZKY, D.S. (Ed.). Advances in NIPS, v. 2, p. 524-532, Morgan Kaufmann, San Mateo, CA, EUA. 1990.

FERNANDES, R.G.; SILVA, D.R.C.; OLIVEIRA, L.A.H.G. de; NETO, A.D.D. Identificação Neural de um Sistema de Níveis em Ambiente Foundation Fieldbus, Congresso Latino Americano de Automática, 2006.

FERNANDES, R.G.; SILVA, D.R.C.; OLIVEIRA, L.A.H.G. de; NETO, A.D.D. Detecção e Isolamento de Falhas em um Sistema de Níveis Real, Simpósio Brasileiro de Automação Inteligente (SBAI), 2007.

FILBERT, D. Fault Diagnosis in Nonlinear Electromechanical Systems by Continuous-time Parameter Estimation. ISA Trans., 24(3):23-27, 1985.

FILBERT, D.; METZGER, L. Quality Test of Systems by Parameter Estimation. In Proc. 9th IMEKO-Congress, Berlim, Alemanha, maio 1982.

FIORIN, D.V; MARTINS, F.R.; SCHUCH, N.J. Aplicações de redes neurais e previsões de disponibilidade de recursos energéticos solares. Revista Brasileira de Ensino de Física, v. 33, n. 1, p. 1309 1 - 1309 20, 2011. https://doi.org/10.1590/S1806-11172011000100009

FOVINO, I. N.; MASERA, M.; CIAN, A. De. Integrating Cyber Attacks Within Fault Trees. Reliability Engineering and System Safety, v. 94, p. 1394-1402, 2009. https://doi.org/10.1016/j.ress.2009.02.020

FRANK, P. Advanced Fault Detection and Isolation Schemes Using Nonlinear and Robust Observers. In 10th IFAC Congress, v. 3, p. 63-68, Munique, Alemanha, 1987. https://doi.org/10.1016/S1474-6670(17)55353-7

FRANK, P. Fault Diagnosis in Dynamic Systems Using Analytical and Knowledge-based Redundancy, Automatica, 26(3):459-474, 1990. https://doi.org/10.1016/0005-1098(90)90018-D

FRANK, P.; WÜNNENBERG, J. Sensor Fault Detection via Robust Observers. In TZAFESTAS, S.; SINGH, M.; SCHMIDT, G., editors, System Fault Diagnostics, Reliability & Related Knowledge-based Approaches, v. 1, p. 147-160, D. Reidel Press, Dordrecht, 1987. https://doi.org/10.1007/978-94-009-3929-5_5

FRIEDMAN, J.H. Another approach to polychotomous classification. Technical report, Department of Statistics, Stanford University, 1996.

FRIES, T.P.; GRAHAM, J. H. A Fuzzy Intelligent Agent Approach to Fault Diagnosis. In: PROCEEDINGS (388) INTELLIGENT SYSTEMS AND CONTROL, 2003

GALANOPOULOS, A.S.; FOWLER, J.E.; AHALT, S.C. Vector Quantization Using Artificial Neural Network. Models Signal Processing in Telecommunications Information Technology: Transmission, Processing and Storage, p. 346-357, 1996. https://doi.org/10.1007/978-1-4471-1013-2_27

GALUSHKIN, A. I. Neural Networks Theory, Springer, 2007.

GEISSER, S. The predictive sample reuse method with applications. Journal of the American Statistical Association, v. 70, p. 320-328, 1975. https://doi.org/10.1080/01621459.1975.10479865

GERTLER, J. A Survey of Model-based Failure Detection and Isolation in Complex Plants. IEEE Control Systems Magazine, 8(6):3-11, 1988. https://doi.org/10.1109/37.9163

GERTLER, J. Fault Detection and Diagnosis in Engineering Systems, Marcel Dekker, New York, 1998.

GERTLER, J.; SINGER, D. Augmented Models for Statistical Fault Isolation in Complex Dynamic Systems. In: Proc. American Control Conference (ACC), v. 1, p. 317-322, Boston, MA, 1985.

GHATE, V.N. DUDUL, S.V. Fault diagnosis of three phase induction motor using neural network techniques, Proc. Second International Conference on Emerging Trends in Engineering & Technology (ICETET), number I, 2009. p. 922-928. https://doi.org/10.1109/ICETET.2009.100

GIL, R.V.; PÁEZ, D.G. Identificación de sistemas dinámicos utilizando redes neuronales RBF. Revista Iberoamericana de Automática e Informática Industrial, v. 4 (2), 2007. p. 32-42. https://doi.org/10.1016/S1697-7912(07)70207-8

GIROSI, F. An equivalence between sparse approximation and Support Vector Machines. A.I. Memo 1606, MIT Artificial Intelligence Laboratory, 1997.

GOMES, G.S.S.; LUDERMIR, T.B. Redes neurais artificiais com funções de ativação complemento log-log e probit para aproximar funções na presença de observações extremas. Learning and Nonlinear Models. Revista da Sociedade Brasileira de Redes Neurais (SBRN), v. 6, n. 2, 2008. p. 142-153. https://doi.org/10.21528/LNLM-vol6-no2-art4

GUARNIERI, S.; PIAZZA, F.; UNCINI, A. Multilayer Feedforward Networks with Adaptive Spline Activation Function. IEEE Transactions on Neural Networks, v. 10 (3), 1999. p. 672-683. https://doi.org/10.1109/72.761726

GUNN, S.R.; BROWN, M.; BOSSLEY, K.M. Network performance assessment for neurofuzzy data modelling. In: LIU, X.; COHEN, P.; Berthold, M. (Eds.), Intelligent Data Analysis, volume 1208, Lecture Notes in Computer Science, p. 313-323, 1997. https://doi.org/10.1007/BFb0052850

GUO, M.-F.; YANG, N.-C. Features-Clustering-Based Earth Fault Detection Using Singular-Value Decomposition and Fuzzy C-Means in Resonant Grounding Distribution Systems, International Journal of Electrical Power & Energy Systems, vol. 93, p. 97-108, 2017. https://doi.org/10.1016/j.ijepes.2017.05.014

GUYON, I.; MATIC, N.; VAPNIK, V.N. Discovering information patterns and data cleaning. MIT Press, Cambridge, p.181-203, 1996

HAGAN, M.T.; MENHAJ, M.B. Training feedforward networks with the Marquardt algorithm. IEEE Transactions on Neural Networks 5 (6), 1994. p. 989-993. https://doi.org/10.1109/72.329697

HAGIWARA, M. A simple and effective method for removal of hidden units and weights. Neurocomputing, v. 6 (2), 1994. p. 207-218. https://doi.org/10.1016/0925-2312(94)90055-8

HASSIBI, B.; STORK, D.G.; WOLF, G.J. Optimal brain surgeon and general network pruning, IEEE ICNN'93, v. 1, 1993. p. 293-299.

HAYKIN, S. Neural Networks. A Comprehensive Foundation. 2ª Edição. Delhi, India: Pearson Prentice Hall. 2005.

HAYKIN, S. Redes Neurais. Princípios e prática. 2ª Edição. São Paulo, Brasil: ARTMED EDITORA S.A., 2007.

HECKMAN, N. The theory and application of penalized least squares methods or reproducing kernel Hilbert spaces made easy, 1997.

HIMMELBLAU, D. Fault Detection and Diagnosis in Chemical and Petrochemical Processes. Elsevier, New York, 1978.

HINTON, G. Supervised learning in supervised neural networks. In: ROBERT, A.W.; KEIL, F.C. (Eds.). The MIT Encyclopedia of the Cognitive Sciences. MIT Press. Cambridge, Massachusetts, London, England. 2001. p. 814-816.

HOCHREITER, S. Untersuchungen zu dynamischen neuronalen Netzen (Diploma thesis), Institut für Informatik, Lehrstuhl Prof. Brauer, Technische Universität München, Advisor: Schmidhuber, J., 1991.

HORNIK, K. Approximation capabilities of multilayer feedforward networks. Neural Networks, v. 4(2), 1991. p. 251-257. https://doi.org/10.1016/0893-6080(91)90009-T

HORNIK, K.; SINCHCOMBE, M.; WHITE, M. Multilayer Feedforward are Universal Approximators. Neural Networks, v. 2, 1989. p. 359-366. https://doi.org/10.1016/0893-6080(89)90020-8

ISERMANN, R. Parameter-adaptive Control Algorithms - A Tutorial. Automatica, 18(5):513-528. 1982. https://doi.org/10.1016/0005-1098(82)90002-4

ISERMANN, R. Process Fault Detection on Modeling and Estimation Methods - A Survey. Automatica, 20(4):387-404, 1984. https://doi.org/10.1016/0005-1098(84)90098-0

ISERMANN, R. Fault Diagnosis of Machines via Parameter Estimation and Knowledge Processing. In Proc. IFAC Symphosium on Fault Detection, Supervision and Safety for Technical Processes (SAFEPROCESS), v. 1, p. 121-133, Baden-Baden, Alemanha, Setembro 1993.

ISERMANN, R. Integration of Fault-detection and Diagnosis Methods. In Proc. IFAC Symposium on Fault Detection, Supervision and Safety for Technical Processes (SAFEPROCESS), p. 597-609, Espoo, Finlândia, Junho 1994.

ISERMANN, R. Supervision, Fault Detection and Fault-Diagnosis Methods - An Introduction. Control Eng. Practice, Vol. 5, No. 5, p. 639-652, 1997. https://doi.org/10.1016/S0967-0661(97)00046-4

ISERMANN, R. Fault-Diagnosis Systems: An Introduction from Fault Detection to Fault Tolerance, Springer, 2006. https://doi.org/10.1007/3-540-30368-5

IVAKHNENKO, A.G. The group method of data handling-a rival of the method of stochastic approximation. Soviet Automatic Control, 13(3), 43-55, 1968.

IVAKHNENKO, A.G. Polynomial theory of complex systems. IEEE Transactions on Systems, Man and Cybernetics, (4), 364-378, 1971. https://doi.org/10.1109/TSMC.1971.4308320

IVAKHNENKO, A.G.; LAPA, V.G. Cybernetic predicting devices. CCM Information Corporation, 1965.

JACQUES, M.A.P.; NIITTYMÄKI, J.; PURSULA, M. Analysing Different Fuzzy Traffic Signal Controllers for Isolated Intersections. Paper accepted for presentation at TRB 81 Annual Meeting. January 13-17, Washington, DC, 2002.

JAIN, A.K.; DUBES, R.C. Algorithms for Clustering Data. New Jersey: Prentice Hall, p. 320, 1988.

JOHANSSON, E.M.; DOWLA, F.U.; GOODMAN, D.M. Backpropagation learning for multilayer feed-forward neural networks using the conjugate gradient method. International Journal of Neural Systems, v. 2 (4), 1991. p. 291-301. https://doi.org/10.1142/S0129065791000261

JONES, H., editor. Failure Detection in Linear Systems. Dept. of Aeronautics, M.I.T., Cambridge, 1973.

JORDAN, M.I. Serial order: a parallel distributed processing approach. Technical report ICS report 8604. San Diego: Institute for Cognitive Science, University of California, 1986.

JORDAN, M.I. Serial order: a parallel distributed processing approach. Advances in Psychology, 121, 471-495, 1997. https://doi.org/10.1016/S0166-4115(97)80111-2

JOSEPH, R.D. Contributions to perceptron theory (Ph.D. thesis), Cornell Univ., 1961.

KASKI, S. Data Exploration Using Self-Organizing Maps, Tese de Doutorado, Escola Politécnica Escandinávia, Finlândia, 1997.

KATIPAMULA, S.; BRAMBLEY, M.R. Methods for Fault Detection, Diagnostics, and Prognostics for Building Systems - A Review, Part I, HVAC&R Research, v. 11, n. 1, janeiro 2005a. https://doi.org/10.1080/10789669.2005.10391123

KATIPAMULA, S.; BRAMBLEY, M.R. Methods for Fault Detection, Diagnostics, and Prognostics for Building Systems - A Review, Part II, HVAC&R Research, v. 11, n. 2, p. 169-187, abril 2005b. https://doi.org/10.1080/10789669.2005.10391133

KEARNS, M. A Bound on the Error of Cross Validation Using the Approximation and Estimation Rates, with Consequences for Information Training-Test Split. In: TOURETZKY, D.S.; MOZER, M.C.; HASSELMO, M.E. (Eds.) Neural Processing 8, Morgan Kaufmann, 1996. p. 183-189.

KENNEDY, P. A guide to econometrics. The MIT Press. Cambridge, MA, EUA. 2003.

KLATT, K.U., ENGELL, S. Gain-scheduling trajectory control of a continuous stirred tank reactor. Computers & Chemical Engineering, p. 491-502, 1998. https://doi.org/10.1016/S0098-1354(97)00261-5

KNERR, S.; PERSONNAZ, L.; DREYFUS, G. Single-layer learning revisited: a stepwise procedure for building and training a neural network. In J. Fogelman, editor, Neurocomputing: Algorithms, Architectures and Applications. Springer-Verlag, 1990.
https://doi.org/10.1007/978-3-642-76153-9_5

KOLEN, J.F.; KREMER, S. Dynamical Recurrent Networks. In: KOLEN, J.F.; KREMER, S.C. (Eds). A field guide to dynamical recurrent neural network. Institute of Electrical and Electronics Engineers, Inc. IEEE Press. New York, NY, EUA. 2001.

KOHONEN, T. Self-Organized Formation of Topologically Correct Feature Map. Biological Cybernetics, v. 43, p. 59-69, 1982.
https://doi.org/10.1007/BF00337288

KOHONEN, T. Self-Organizing Map. Proceedings of the IEEE, v. 78, n. 9, p. 1464-1480, 1990.
https://doi.org/10.1109/5.58325

KOHONEN, T. Self-Organizing Maps. 2 ed. Springer, 1997.
https://doi.org/10.1007/978-3-642-97966-8

KOSCIELNY, J. M.; SYFERT, M.; BARTYS, M. Fuzzy-Logic Fault Diagnosis of Industrial Process Actuators, Journal of Applied Mathematics and Computer Science, 1999.

KRESSEL, U. H.-G. Pairwise classification and support vector machines. In SCHOLKOPF, B.; BURGES, C. J. C.; SMOLA, A. J., editors, Advances in Kernel Methods - Support Vector Learning, p. 255-268, Cambridge: The MIT Press, 1998.

KRÖSE, B.; VAN DER SMAGT, P. An Introduction to Neural Networks. 8. ed. Amsterdam: The University of Amsterdam, 1996.

LAHMIRI, S. A Comparative Study of Backpropagation Algorithms in Financial Prediction. International Journal of Computer Science, Engineering and Applications(IJCSEA), v. 1 (4), 2001. DOI: 10.5121/ijcsea.2011.1402.
https://doi.org/10.5121/ijcsea.2011.1402

LECUN, Y. Une procédure d'apprentissage pour réseau à seuil asymétrique. In Proceedings of cognitiva 85, p. 599-604, 1985.

LECUN, Y. A theoretical framework for back-propagation. In TOURETZKY, D.; HINTON, G.; SEJNOWSKI, T. (Eds.), Proceedings of the 1988 Connectionist Models Summer School, p. 21-28. CMU, Pittsburgh, Pa: Morgan Kaufmann, 1988.

LECUN, Y.; DENKER, J.; SOLLA, S.; HOWARD, R.E.; JACKEL, L.D. Optimal brain damage. In: TOURETZKY, D.S. (Ed.); Advances in NIPS, v. 2, Morgan Kaufmann, San Mateo, CA. 1990.

LEE, H.; GROSSE, R.; RANGANATH, R.; NG, A.Y. Convolutional deep belief networks for scalable unsupervised learning of hierarchical representations. In Proceedings of the 26th International Conference on Machine Learning, p.609-616, 2009a. https://doi.org/10.1145/1553374.1553453

LEE, H.; PHAM, P.T., LARGMAN, Y.; NG, A.Y. Unsupervised feature learning for audio classification using convolutional deep belief networks. In Proc. NIPS, vol. 9, p. 1096-1104, 2009b.

LESHNO, M.; LIN, V.Y.; PINKUS, A.; SCHOCKEN, S. Multilayer feedforward networks with a nonpolynomial activation function can approximate any function. Neural Networks, v. 6 (6), 1993. p. 861-867. https://doi.org/10.1016/S0893-6080(05)80131-5

LI, L. Fault Detection and Fault-Tolerant Control for Nonlinear Systems, Springer Vieweg, 2016. https://doi.org/10.1007/978-3-658-13020-6

LI, L.; DING, S. X.; YANG, Y.; ZHANG, Y. Robust Fuzzy Observer-based Fault Detection for Nonlinear Systems with Disturbances, Neurocomputing, v. 174, 2016. p. 767-772. https://doi.org/10.1016/j.neucom.2015.09.102

LI, Z.; CHEN, Z.. Research on Genetic Algorithm Fault Diagnosis in Chemical Process, IOP Conf. Ser.: Mater. Sci. Eng. 563, 052007, 2019. https://doi.org/10.1088/1757-899X/563/5/052007

LIN, C.-F.; WANG, S.-D. Fuzzy Support Vector Machines, in: IEEE Transactions on Neural Networks, vol. 13, n. 2, mar. 2002. https://doi.org/10.1109/72.991432

LINARIÉ, D.; KOROMAN, V. Fault Diagnosis of a Hydraulic Actuator Using Neural Network. In: IEEE, ICIT / KAREL JEZERNIK, 2003.

LIU, J.; ZIO, E. A Scalable Fuzzy Support Vector Machine for Fault Detection in Transportation Systems, Expert Systems with Applications, vol. 102, pp. 36-43, 2018. https://doi.org/10.1016/j.eswa.2018.02.017

LJUNG, L. System identification. In: LEVINE, W.S. (Ed.). The Control Handbook, CRC Press Inc., 1996. p. 1033-1054.

LJUNG, L. System Identification: Theory for the user, Prentice-Hall, Engledwood Cliffs, NJ, 1987.

LOU, S. J.; BUDMAN, H.; DUEVER, T. A. Comparison of Fault Detection Techniques, Journal of Process Control, v. 13, 2003. p. 451-464. https://doi.org/10.1016/S0959-1524(02)00069-0

MA, J.; CAO, B. The Mahalanobis Distance Based Rival-Penalize Competitive Learning Algorithm. In: WANG, J.; YI, Z.; ZURADA, J.M.; LU, B.; YIN, H. Advances in Neural Networks. Berlin-Heilderberg: Springer-Verlag, p. 442-447, 2006. https://doi.org/10.1007/11759966_66

MAASS, W. On The Computational Power of Winner-Take-All. Electronic Colloquium on Computational Complexity (ECCC), n. 32, p. 1-19, 2000.

MADANI, K. A Survey of Artificial Neural Networks Based Fault Detection and Fault Diagnosis Techniques, Proceedings of IEEE, 1999.

MALONE, J.; MCGARRY, K.; WERMTER, S.; BOWERMAN, C. Data Mining Using Rule Extraction from Kohonen Self-Organizing Maps. Neural Computing & Applications, v. 15, n. 1, p. 9-17, 2006. https://doi.org/10.1007/s00521-005-0002-1

MANDIC, D.P.; CHAMBERS, J.A. Recurrent Neural Networks for Prediction. Chichester, Inglaterra: John Wiley & Sons, Inc. 2001. https://doi.org/10.1002/047084535X

MARTINETZ, T.; SCHULTEN, K.A. "Neural Gas" Network Learns Topologies. In: KOHONEN, T.; MÄKISARA, K.; SIMULA, O.; KANGAS, J. Editors, Artificial Neural Networks. Amsterdam: Elsevier, p. 397-402, 1991.

MCCULLOCH, W.S.; PITTS, W. A Logical Calculus of the Ideas Immanent in Nervous Activity. Bulletin of Mathematical Biophysics, v. 5, 1943. p. 115-133. https://doi.org/10.1007/BF02478259

MEHRA, R.; PESCHON, J. An Innovations Approach to Fault Detection and Diagnosis in Dynamic Systems. Automatica, 7:637-640, 1971 https://doi.org/10.1016/0005-1098(71)90028-8

MEHROTRA, K.; MOHAN, C.K.; RANKA, S. Elements of Artificial Neural Networks. Cambridge: The MIT Press, p. 329, 1996. https://doi.org/10.7551/mitpress/2687.001.0001

MENDONCA, L.F.; SOUSA, J.M.C.; SA DA COSTA, J.M.G. Fault Detection and Isolation of Industrial Processes using Optimized Fuzzy Models, IEEE International Conference on Fuzzy Systems, 2005. https://doi.org/10.1109/FUZZY.2005.1452505

MENNON, A.; MEHROTRA, K.; MOHAN, C.K.; RANKA, S. Characterization of a class of sigmoid functions with applications to neural networks. Neural Networks, 9(5), 1996. p. 819-835. https://doi.org/10.1016/0893-6080(95)00107-7

MENNON, A.; MEHROTRA, K.; MOHAN, C.K.; RANKA, S. Elements of artificial neural networks. Cambridge: The MIT Press, 2000.

MESKIN, N.; KHORASANI, K. Fault Detection and Isolation, Springer, 2011. https://doi.org/10.1007/978-1-4419-8393-0

MILJKOVIĆ, D. Fault Detection Methods: A Literature Survey, MIPRO, 110-115, 2011.

MING, L.; ZHAO, J. Review on Chemical Process Fault Detection and Diagnosis. In: 6TH INTERNATIONAL SYMPOSIUM ON ADVANCED CONTROL OF INDUSTRIAL PROCESSES (ADCONIP 2017), Taipei, Taiwan, 2017. https://doi.org/10.1109/ADCONIP.2017.7983824

MINOUX, M. Mathematical Programming: Theory and Algorithms. John Wiley & Sons, 1986.

MINSKY, M. Steps toward artificial intelligence. In FEIGENBAUM, E.; FELDMAN, J. (Eds.), Computers and thought, p. 406-450. New York: McGraw-Hill, 1963.

MOBILIS. Introdução aos Self Organizing Maps. Disponível em: <http://www.decom.ufop.br/imobilis/self-organizing-maps/> Acesso em: 20 de junho de 2018.

MONSEF, H.; RANJBAR, A. M.; JADID, S. Fuzzy Rule-based Expert System for Power System Fault Diagnosis. In: IEE PROCEEDINGS - GENERATION, TRANSMISSION AND DISTRIBUTION, 1997. https://doi.org/10.1049/ip-gtd:19970799

MONTGOMERY, D.C. Design and Analysis of Experiments. 5ª Ed., John Wiley & Sons, New York, NY, EUA. 2001.

MORENO-ARMENDARIZ, M.A.; POZNYAK, A.S.; YU-LIU, W. Control adaptable indirecto usando redes neuronales dinámicas. Revista del Centro de Investigación. Universidad La Salle, México, v. 4 (16), 2001. p. 13-32.

MOZER, M.C.; SMOLENSKY, P. Skeletonization: A Technique for Trimming the Fat From a Network Via Relevance Assessment. In: TOURETZKY, D.S. (Ed.). Advances in Neural Information Processing, v. 1, p. 107-116. San Mateo, CA. Morgan Kaufmann. 1989. https://doi.org/10.1080/09540098908915626

MSA TECHNOSOFT. What is Neural Network? Disponível em: < https://msatechnosoft.in/blog/tech-blogs/artificial-neural-network-types-feed-forward-feedback-structure-perceptron-machine-learning-applications> Acesso em: 11 de outubro de 2018.

NAIR, T.M.; ZHENG, C.L.; FINK, J.L.; STUART, R.O.; GRIBSKOV, M. Rival Penalized Competitive Learning (RPCL): a Topology-determining Algorithm for Analyzing Gene Expression Data, Computational Biology and Chemistry, v. 27, p. 565-574,

2003.
https://doi.org/10.1016/j.compbiolchem.2003.09.006

NASCIMENTO, C.L.; YONEYAMA, T. Inteligência artificial em controle e automação. Blücher-FAPESP. São Paulo, Brasil. 2008.

NAUCK, D.; KRUSE, R. NEFCLASS - A Neuro-Fuzzy Approach for the Classification of Data, ACM Symposium on Applied Computing. Nashville: ACM Press, Feb 26-28, p. 461-465, 1995.

NAUGHTON, J. M.; CHEN, Y. C.; JIANG, J. A Neural Networks Application to Fault Diagnosis for Robotic Manipulator. In: RYERSON POLYTECHNIC UNIVERSITY TORONTO AND UNIVERSITY OF WESTERN ONTARIO LONDON, 1996.

OGUNNAIKE, B.A.; RAY, W.H. Process Dynamic Modeling and Control. New York, NY, EUA: Oxford University Press. Oxford. 1994.

OLIVEIRA, J. C. M.; PONTES, K. V.; SARTORI, I.; EMBIRUÇU, M. Fault Detection and Diagnosis in Dynamic Systems Using Weighless Neural Networks, Expert Systems With Applications, v. 84, 2017. p. 200-219.
https://doi.org/10.1016/j.eswa.2017.05.020

OLIVIER, L. E.; CRAIG, I. K. Should I Shut Down my Processing Plant? An Analysis in the Presence of Faults, Journal of Process Control, v. 56, 2017. p. 35-47.
https://doi.org/10.1016/j.jprocont.2017.05.005

OSTER, M.; DOUGLAS, R.; LIU, S.C. Computation With Spikes in a Winner-Take-All Network, Neural Computation, v. 21, n. 9, p. 2437-2465, 2009.
https://doi.org/10.1162/neco.2009.07-08-829

PALAMARA, F.; PIGLIONE, F.; PICCININI, N. Self-Organizing Map and Clustering Algorithms for the Analysis of Occupational Accident Databases. Safety Science, v. 49, n. 8-9, p. 1215-1230, 2011.
https://doi.org/10.1016/j.ssci.2011.04.003

PAN, Y.; LI, H.; ZHOU, Q. Fault Detection for Interval Type-2 Fuzzy Systems with Sensor Nonlinearities, Neurocomputing, vol. 145, p. 488-494, 2014.
https://doi.org/10.1016/j.neucom.2014.05.005

PARKER, D.B. Learning-logic. Technical report TR-47. Center for Comp. Research in Economics and Management Sci., MIT, 1985.

PATTON, R. Robust Fault Detection Using Eigenstructure Assignment. In Proc. 12th IMACS World Congress on Scientific Computation, v. 2, p. 431-434, Paris, França, 1988.

PATTON, R. Robust Model-based Fault Diagnosis: The State of The Art. In Proc. IFAC Symposium on Fault Detection, Supervision and Safety for Technical Processes (SAFEPROCESS), p. 1-24, Helsinque, Finlândia, 1994.

PAU, L., editor. Failure Diagnosis and Performance Monitoring, Contr. & Syst. Theory Series, v. 11. Marcel Dekker, New York, 1975.

PEDRYCZ, W. Logic-based neurons: Extensions, uncertainty representation and development of fuzzy controlers. Fuzzy Sets and Systems, 66, p. 251-266, 1994. https://doi.org/10.1016/0165-0114(94)90093-0

PRECHELT, L. Automatic early stopping using cross validation: quantifying the Criteria. Neural Networks, v. 11 (4), June, p. 761-767. 1998a. https://doi.org/10.1016/S0893-6080(98)00010-0

PRECHELT, L. Early stopping - but when? In: ORR, G.; MÜLLER, K.R. (Eds.). Neural Networks: Tricks of the Trade, Lecture Notes in Computer Science, v. 1524, (2), p. 55-69, Springer-Verlag. 1998b. https://doi.org/10.1007/3-540-49430-8_3

RAGHAVARAO, D.; PADGETT, L. Blocks Designs: Analysis, Combinatorics, and Applications. Cingapura: World Scientific Publishing. 2005. https://doi.org/10.1142/5846

RAINA, R., MADHAVAN, A., NG, A. Large-scale deep unsupervised learning using graphics processors. In Proceedings of the 26th annual International conference on machine learning, p. 873-880. ACM, 2009. https://doi.org/10.1145/1553374.1553486

RANGANATHAN, A. The Levenberg-Marquardt Algorithm. Disponível em: <http://users-phys.au.dk/jensjh/numeric/project/10.1.1.135.865.pdf>. Acesso em: 21 novembro 2013. 2004.

RANKOVIC, V.M.; NIKOLIC, I.Z. Identification of Nonlinear Models with Feedforward Neural Network and Digital Recurrent Network. FME Transactions, v. 36 (2), 2008. p. 87-92.

REED, R. Pruning Algorithms - A Survey. IEEE Transactions on Neural Networks, v. 4 (5), 1993. p. 740-747. https://doi.org/10.1109/72.248452

REITERMANOVA, Z. Feedforward Neural Networks-Architecture Optimization and Knowledge Extraction. WDS'08 Proceedings of Contributed Papers, Part I, 2008. p. 159-164.

RICH, E.; KNIGHT, K. Inteligência Artificial, São Paulo: Makron Books, 1994.

RITTER, H.; KOHONEN, T. Self-Organizing Semantic Maps. Biological Cybernetics, v. 61, n. 4, p. 241-254, 1989. https://doi.org/10.1007/BF00203171

ROCHA, R.R. Controle Preditivo Distribuído para Sistemas Não Lineares com Particionamento Automático. Tese de doutorado apresentada ao PPGEQ-UFU. Uberlândia, 2018.

RUMELHART, D.E.; HINTON, G.E.; WILLIAMS, R.J. Learning representations by back propagating errors. Nature, v. 323, pp. 533-536. 1986. https://doi.org/10.1038/323533a0

RUMELHART, D.E.; ZIPSER, D. Feature Discovery by Competitive Learning. In: RUMELHART, D.E.; MCLELAND, J.L. PDP Research Group, Parallel Distributed Processing Vol. 1 - Explorations in The Microestructure of Cognition: Foundations. The MIT Press, Cambridge, p. 151-193, 1986. https://doi.org/10.7551/mitpress/5236.001.0001

RUSSEL, E.L.; CHIANG, L.H.; BRAATZ, R.D. Data-driven Techniques for Fault Detection and Diagnosis in Chemical Processes, London: Springer, 2000.

SALAHSHOOR, K.; KORDESTANI, M.; KHOSHRO, M.S. Fault detection and diagnosis of an industrial steam turbine using fusion of SVM (support vector machine) and ANFIS (adaptive neuro-fuzzy inference system) classifiers. Energy, 35, p. 5472-5482, 2010. https://doi.org/10.1016/j.energy.2010.06.001

SALATAS, J. Implementation of Competitive Learning Networks for WEKA. 2011. Disponível em: <http://jsalatas.ictpro.gr/implementation-of-competitive-learning-networks-for-weka/>. Acesso em: 25 de Agosto de 2018.

SAMANTARAY, S.R.; DASH, P.K.; UPADHYAY, S.K. Adaptive Kalman filter and neural network based high impedance fault detection in power distribution networks, International Journal of Electrical Power & Energy Systems, Vol. 31, nº 4, p. 167-172, 2009. https://doi.org/10.1016/j.ijepes.2009.01.001

SANTOS, R.J.; BAUCHSPIESS, A.; BORGES, G.A. Laboratório remote de automação predial. In: XXXII Congresso Brasileiro de Ensino da Engenharia, COBENGE 2004, Brasília, DF, 14 - 17 setembro, 2004. Anais. Brasília: UnB, 2004.

SCHMIDHUBER, J. Dynamische neuronale Netze und das fundamentale raumzeitliche Lernproblem. (Dynamic neural nets and the fundamental spatiotemporal credit assignment problem.) (Dissertation), Inst. f. Inf., Tech. Univ. Munich, 1990.

SCHÖLKOPF, B.; BURGES, C.J.C.; SMOLA, A.J. (Eds.) Advances in kernel methods-support vector learning. Cambridge, MA: MIT Press, 1998.

SCHÖLKOPF, B.; SMOLA, A.J. Leaning with kernels - support vector machines, regularization, optimization, and beyond. The MIT Press, 2002.

SHAO, J. Linear model selection by cross-validation. J. Amer. Statist. Assoc., 88 (422), 1993. p. 486-494. https://doi.org/10.1080/01621459.1993.10476299

SHARMA, S.K.; CHANDRA, P. Constructive neural networks: a review. International Journal of Engineering Science and Technology. 2 (12), 2010. p. 7847-7855.

SHEN, B.; WANG, Z.; DONG, H.; LI, Q. An H∞ Disturbance Estimation Approach to Fault Detection for Linear Discrete Time-Varying Systems, IFAC-PapersOnLine, v. 48 (28), 2015. p. 857-862.

SIEBERT, H.; ISERMANN, R. Leckerkennung und -lokalisierung bei Pipelines durch Online-Korrelation mit einen Prozeβrechner. Regelungstechnik, 25(3):69-74, 1977. https://doi.org/10.1524/auto.1977.25.112.69

SIEGELMANN, H.T.; SONTAG, E.D. Turing computability with neural nets. Applied Mathematics Letters, 4(6), p. 77-80, 1991 https://doi.org/10.1016/0893-9659(91)90080-F

SILVA, D.R.C. Sistema de Detecção e Isolamento de Falhas em Sistemas Dinâmicos Baseado em Identificação Parâmetrica, Tese de Doutorado, UFRN, Natal-RN, 2008.

SILVA, D.R.C.; FERNANDES, R.G.; OLIVEIRA, L.A. H.G. de; NETO, A.D.D. An Implementation of a Fault Detection and Isolation System on Foundation Fieldbus Environment, Emerging Technologies, Robotics and Control Systems, v. 2, 2007. p. 62-68.

SILVA, F.L.H. Modelagem, simulação e controle de fermentação alcoólica continua extrativa. Campinas. Tese de Doutorado. Faculdade de Engenharia de Alimentos, Universidade Estadual de Campinas, Brasil. 1998.

SIMANI, S. Identification and Fault Diagnosis of a Simulated Model of an Industrial Gas Turbine. In: IEEE TRANSACTIONS ON INDUSTRIAL INFORMATICS, v. 1, 2005. p. 202-216. https://doi.org/10.1109/TII.2005.844425

SIMÕES, M.G.; SHAW, I.S. Controle e Modelagem Fuzzy. 2ª Edição. São Paulo, Brasil: Blücher-FAPESP, 2007.

SIVARAM, A.; DAS, L.; VENKATASUBRAMANIAN, V. Hidden representations in deep neural networks: Part 1. Classification problems. Computers & Chemical Engineering, v. 134, p. 106669, 2020. https://doi.org/10.1016/j.compchemeng.2019.106669

SMOLA, A.J. Regression Estimation with Support Vector Learning Machines. Master's thesis, Technische Universität München, 1996.

SMOLA, A.J.; SCHÖLKOPF, B. On a kernel-based method for pattern recognition, regression, approximation and operator inversion. Algorithmica, 22:211-231, 1998. Technical Report 1064, GMD FIRST, Abril 1997. https://doi.org/10.1007/PL00013831

SOBHANI-TEHRANI, E.; KHORASANI, K. Fault Diagnosis of Nonlinear Systems Using a Hybrid Approach, Lecture Notes in Control and Information Science, Springer, 2009. https://doi.org/10.1007/978-0-387-92907-1

SOLOWAY, E. Learning to program = learning to construct mechanisms and explanations. Communications of the ACM, 29(9), p. 850-858, 1986. https://doi.org/10.1145/6592.6594

SRINIVASAN, A.; BATUR, C. Hopfield/art-1 Neural Network-Based Fault Detection and Isolation. In: IEEE TRANSACTIONS ON NEURAL NETWORKS, v. 5, 1994. https://doi.org/10.1109/72.329685

STAHLBERGER, A.; RIEDMILLER, M. Fast network pruning and feature extraction by using the unit-OBS algorithm. In: MOZER, M.C.; JORDAN, M.I.; Petsche, T. (Eds.). Advances in Neural Information Processing Systems, v. 9, p. 655. The MIT Press. 1997.

STEWART, B.T.; WRIGHT, S.J.; RAWLINGS, J.B. Cooperative distributed model predictive control for nonlinear systems. Journal of Process Control, v. 21, n. 5, p. 698 - 704, 2011. https://doi.org/10.1016/j.jprocont.2010.11.004

SUTTON, R.S.; BARTO, A.G. Reinforcement Learning: An Introduction. Cambridge: The MIT Press, p. 322, 1998.

TAHMASEBI, P.; HEZARKHANI, A. Comparison of Optimized Neural Network with Fuzzy Logic for Ore Grade Estimation. Australian Journal of Basic & Applied Sciences, v. 15 (1), 2010. p. 116-132.

THOMASSEY, S.; HAPPIETTE, M. A Neural Clustering And Classification System for Sales Forecasting of New Apparel Items, Appl. Soft Comput., v. 7, p. 1177-1187, 2007. https://doi.org/10.1016/j.asoc.2006.01.005

TINOS, R.; TERRA, M.H. Fault Detection and Isolation in Robotic Manipulators and the Radial Basis Function Network Trained by the Kohonen's Self-Organizing Map. In: SBRN, 1998. p. 85-90.

ULTSCH, A. U*-Matrix: a Tool to visualize Clusters in high dimensional Data. University of Maburg, Dep. of Comp. Sci., Technical Report, n. 36, December 2003, Disponível

em: < https://pdfs.semanticscholar.org/1d9d/ba44f2d237ee9d8f0388299afbcc0e581621.pdf > Acesso em: 15 de julho de 2018.

VAN OVERSCHEE, P. D. M. B. N4Sid: Subspace algorithms for the identification of combined deterministic-stochastic system, Automatica, v. 30 (1), 1994. p. 75-93. https://doi.org/10.1016/0005-1098(94)90230-5

VAPNIK, V. Statistical Learning Theory. Springer, N.Y., 1998.

VAPNIK, V. The Nature of Statistical Learning Theory. Springer, N.Y., 1995. ISBN 0-387-94559-8. https://doi.org/10.1007/978-1-4757-2440-0

VAPNIK, V.; GOLOWICH, S.; SMOLA, A. Support vector method for function approximation, regression estimation, and signal processing. In: MOZER, M.; JORDAN, M.; PETSCHE, T. (Eds.), Advances in Neural Information Processing Systems 9, pages 281-287, Cambridge, MA, 1997. MIT Press.

VECCI, L.; PIAZZA, F.; UNCINI, A. Learning and approximation capabilities of adaptive spline activation function neural networks. Neural Networks, v. 11 (2), 1998. p. 259-270. https://doi.org/10.1016/S0893-6080(97)00118-4

VENKATASUBRAMANIAN, V.; RENGASWAMY, R.; YIN, K.; KAVURI, S.N. A Review of Process Fault Detection and Diagnosis Part I: Quantitative model-based methods, Computers and Chemical Engineering (27), p. 293-311, 2003a. https://doi.org/10.1016/S0098-1354(02)00160-6

VENKATASUBRAMANIAN, V.; RENGASWAMY, R.; KAVURI, S.N. A Review of Process Fault Detection and Diagnosis Part II: Qualitative model-based methods, Computers and Chemical Engineering (27), p. 313-326, 2003b. https://doi.org/10.1016/S0098-1354(02)00161-8

VENKATASUBRAMANIAN, V.; RENGASWAMY, R.; KAVURI, S.N.; YIN, K.; A Review of Process Fault Detection and Diagnosis Part III: Process history based methods, Computers and Chemical Engineering (27), p. 327-346, 2003c. https://doi.org/10.1016/S0098-1354(02)00162-X

VIGLIONE, S. Applications of pattern recognition technology. In MENDEL, J.M.; FU, K.S. (Eds.), Adaptive, learning, and pattern recognition systems. Academic Press, 1970. https://doi.org/10.1016/S0076-5392(08)60492-0

WAHBA, G. Spline Models for Observational Data. Series in Applied Mathematics, Vol. 59, SIAM, Philadelphia, 1990. https://doi.org/10.1137/1.9781611970128

WANG, S.; ZHANG, J., "An Intelligent Process Fault Diagnosis System Integrating Andrews Plot, PCA and Neural Networks," 2019 25th International Conference on Automation and Computing (ICAC), Lancaster, United Kingdom, 2019, pp. 1-6. https://doi.org/10.23919/IConAC.2019.8895115

WATANABE, U.; HIMMELBLAU, D. Instrument Fault Detection in Systems With Uncertainties. Int. Journal of Systems Science, 13:137-158, 1982. https://doi.org/10.1080/00207728208926337

WERBOS, P. J. Applications of advances in nonlinear sensitivity analysis. In Proceedings of the 10th IFIP conference, 31.8-4.9, NYC, p. 762-770, 1981. https://doi.org/10.1007/BFb0006203

WERBOS, P. J. Backwards differentiation in AD and neural nets: Past links and new opportunities. In Automatic differentiation: applications, theory, and implementations, p. 15-34, Springer, 2006. https://doi.org/10.1007/3-540-28438-9_2

WILLSKY, A. A Survey of Design Methods for Failure Detection Systems. Automatica, 12:601-611, 1976. https://doi.org/10.1016/0005-1098(76)90041-8

WU, Y.; DONG, J. Fault Detection for T-S Fuzzy Systems with Partly Unmeasurable Premise Variables, Fuzzy Sets and Systems, vol. 338, p. 136-156, 2018. https://doi.org/10.1016/j.fss.2017.06.006

WU, H.; ZHAO, J. Deep convolutional neural network model based chemical process fault diagnosis. Computers & Chemical Engineering, v. 115, p. 185-197, 2018. https://doi.org/10.1016/j.compchemeng.2018.04.009

XU, D.; TIAN, Y. A Comprehensive Survey of Clustering Algorithms. Ann. Data. Sci., v. 2, n. 2, p. 165-193, 2015. https://doi.org/10.1007/s40745-015-0040-1

XU, L.; KRZYZAK, A.; OJA, E. Rival Penalized Competitive Learning For Clustering Analysis, RBF Net, and Curve Detection. IEEE Transactions on Neural Networks, v. 4, n. 4, p. 636-649, 1993. https://doi.org/10.1109/72.238318

YAHYA, M.N.; OTSURU, T.; TOMIKU, R.; OKOZONO, T. Investigation the capability of neural network in predicting reverberation time on classroom. International Journal of Sustainable Construction Engineering & Technology, v. 1 (1), 2010. p. 1-13.

YANG, X.; WANG, W.; LUO, Y.-W. Binary Tree SVM Based on Analytic Hierarchy Process and Its Application to Fault Diagnosis, DEStech Transactions on Engineering and Technology Research, 2019. https://doi.org/10.12783/dtetr/amee2019/33432

YEGNANARAYANA, B. Tutorial on Neural Network Models for Speech and Image Processing, Speech & Vision Laboratory, Dept. of Computer Science & Engineering, IIT Madras, WCCI 2002, Honolulu, Hawaii, USA, 2002.

ZADEH, L.A. Fuzzy sets, Information and Control, Volume 8, Issue 3, June, 1965, p. 338-353.
https://doi.org/10.1016/S0019-9958(65)90241-X

ZADEH, L.A. Outline of a New Approach to the Analysis of Complex Systems and Decision Processes, IEEE Trans. on Systems, Man, and Cybernetics, v. SMC-3, n. 1, p. 28-44, Jan. 1973.
https://doi.org/10.1109/TSMC.1973.5408575

ZHANG, X.G. Using class-center vectors to build support vector machines. In: IEEE proceedings of the neural networks and signal processing IX, Aug 1999.

ZHANG, F.; GE, Z. Decision Fusion Systems for Fault Detection and Identification in Industrial Processes, Journal of Process Control, 2015.
https://doi.org/10.1016/j.jprocont.2015.04.004

ZHANG, M., WANG, R., CAI, Z., CHENG, W.. Phase partition and identification based on kernel entropy component analysis and multi-class support vector machines-fireworks algorithm for multi-phase batch process fault diagnosis. Transactions of the Institute of Measurement and Control, 2020.
https://doi.org/10.1177/0142331220910885

APÊNDICE A

Os resultados obtidos para o modelo de Stewart e colaboradores (2011), estão listados neste Apêndice, divididos de acordo com a metodologia de detecção de falhas, entre elas o SVC OAA, SVC OAO, ANN FF, ANN WTA, ANN SOM e *Fuzzy*.

A.1. Resultados SVC OAA

Foram estabelecidas 24 falhas, sendo duas falhas para cada variável, aplicadas em cada um dos dois estados estacionários estipulados, de acordo com o Capítulo 3. Além das 24 falhas, foram utilizados dados de ambos os estados estacionários, totalizando 26 situações passíveis de detecção. A metodologia de detecção de falhas SVM *one against all* utiliza o conjunto de dados e rótulos estabelecidos para os sinais de treinamento de forma a criar hiperplanos ótimos que separam as diferentes classes (falhas e operações normais) umas das outras. A metodologia *one against all* gera modelos que separam uma classe de todas as outras classes existentes, resultando para este caso 26 modelos diferentes.

Para todas as situações passíveis de detecção, foram simulados computacionalmente 26 conjuntos de sinais dispostos em uma matriz, em que outro estado estacionário ou falha foram aplicados nos instantes 5 minutos e 45 minutos, em um total de 60 minutos de operação, para cada sinal treinado. Os dados então foram separados em dados de treinamento e de validação. Foi utilizada uma proporção de 2 para 1 entre os dados de treinamento e validação. A cada dois instantes separados para treinamento, o terceiro instante era designado para validação. Foram estabelecidos para os dados de treinamento e validação rótulos para cada classe de dados, de acordo com a Tabela A.1.

Tabela A.1 - Rótulos utilizados para detecção de falhas utilizando SVC.

Estado	Amplitude	Rótulo	Estado	Amplitude	Rótulo
Q_3 (ee1)	0,70	1	ee2	-	15
Q_2 (ee1)	0,70	2	H_1 (ee2)	0,70	16
Q_1 (ee1)	0,70	3	H_2 (ee2)	0,70	17
F_R (ee1)	1,30	4	H_3 (ee2)	0,70	18
F_{f2} (ee1)	1,30	5	T_1 (ee2)	1,02	19
F_{f1} (ee1)	1,30	6	T_2 (ee2)	1,02	20
T_3 (ee1)	0,98	7	T_3 (ee2)	1,02	21
T_2 (ee1)	0,98	8	F_{f1} (ee2)	0,70	22
T_1 (ee1)	0,98	9	F_{f2} (ee2)	0,70	23
H_3 (ee1)	1,30	10	F_R (ee2)	0,70	24
H_2 (ee1)	1,30	11	Q_1 (ee2)	1,30	25
H_1 (ee1)	1,30	12	Q_2 (ee2)	1,30	26
ee1	-	13	Q_3 (ee2)	1,30	27
0	-	14	-	-	-

Após o treinamento do sistema de detecção de falhas, foram simulados um novo conjunto de dados, em que a falha (ou outro estado estacionário) eram aplicados nos instantes 10 minutos e 35 minutos, mantendo o total de 60 minutos de operação para os sinais.

Nas figuras a seguir, apresentam-se os resultados obtidos para o sistema de detecção, para todas as situações apresentadas na Tabela 3.5.

Figura A.1 - Detecção SVC OAA - Op. *ee*2 - *ee*1 - *ee*2.

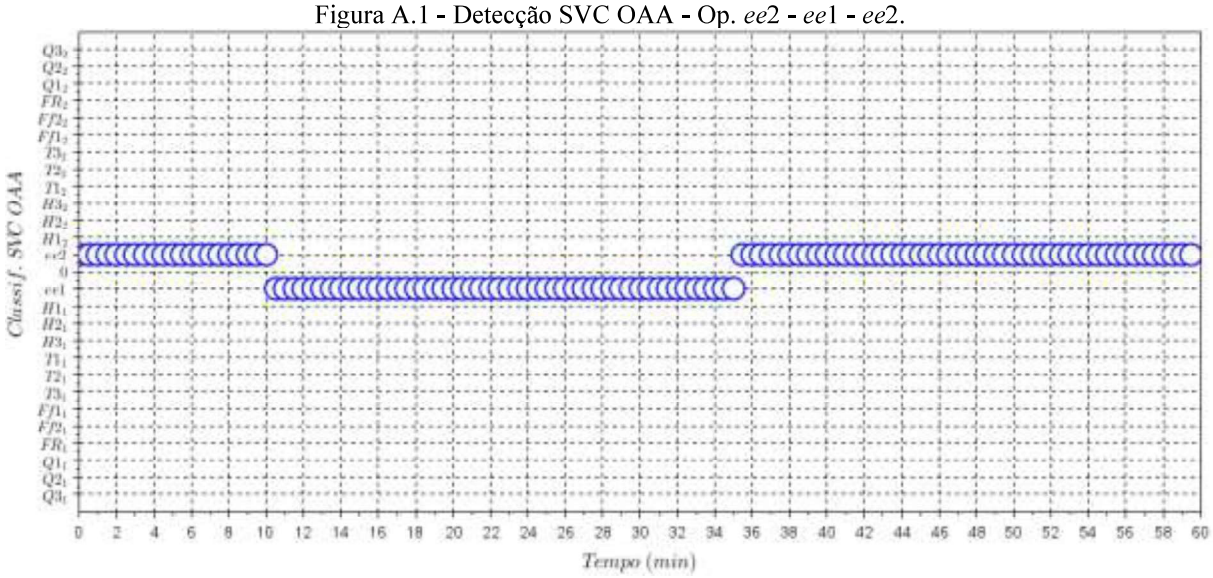


Figura A.2 - Detecção SVC OAA - Op. *ee*1 - *ee*2 - *ee*1.

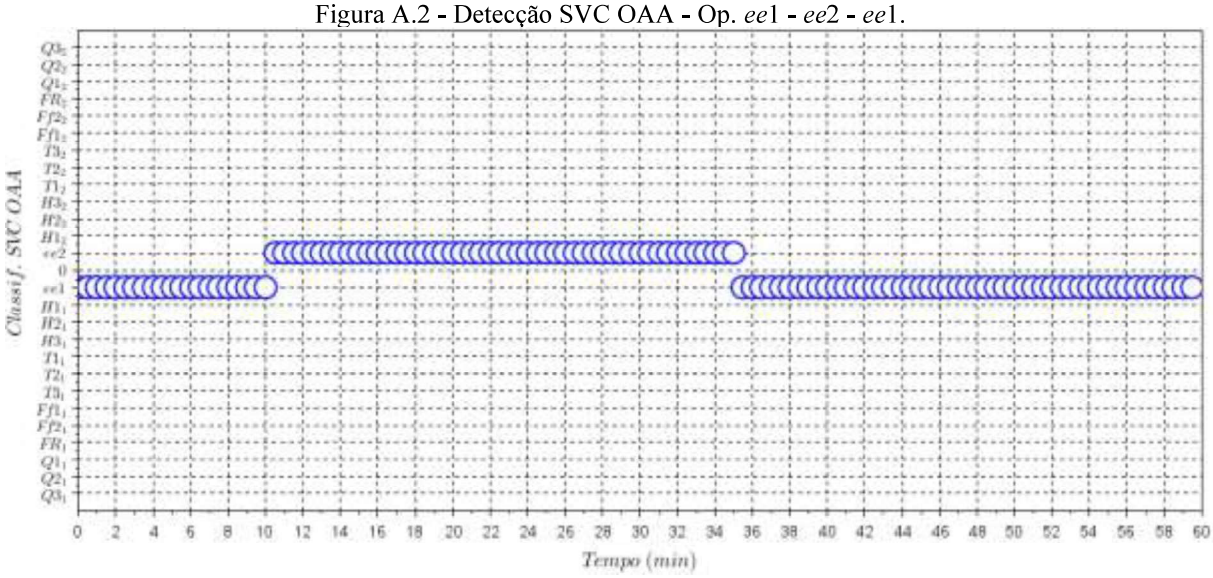


Figura A.3 - Detecção SVC OAA - Op. *ee*2 - *ee*1 - falha $H1_1$.

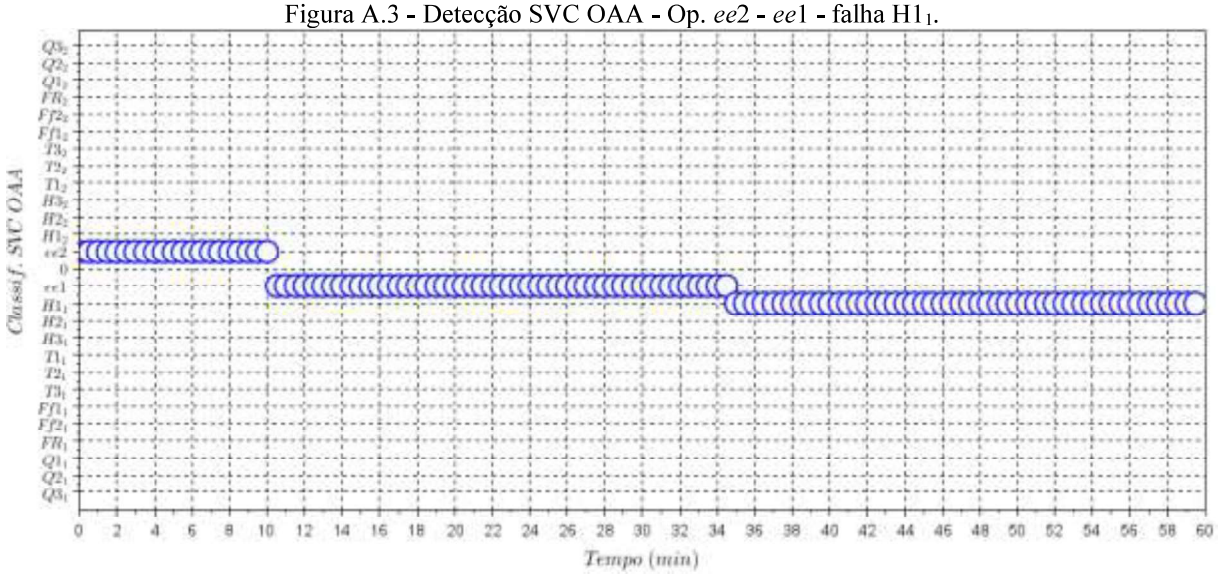


Figura A.4 - Detecção SVC OAA - Op. *ee*1 - *ee*2 - falha $H1_2$.

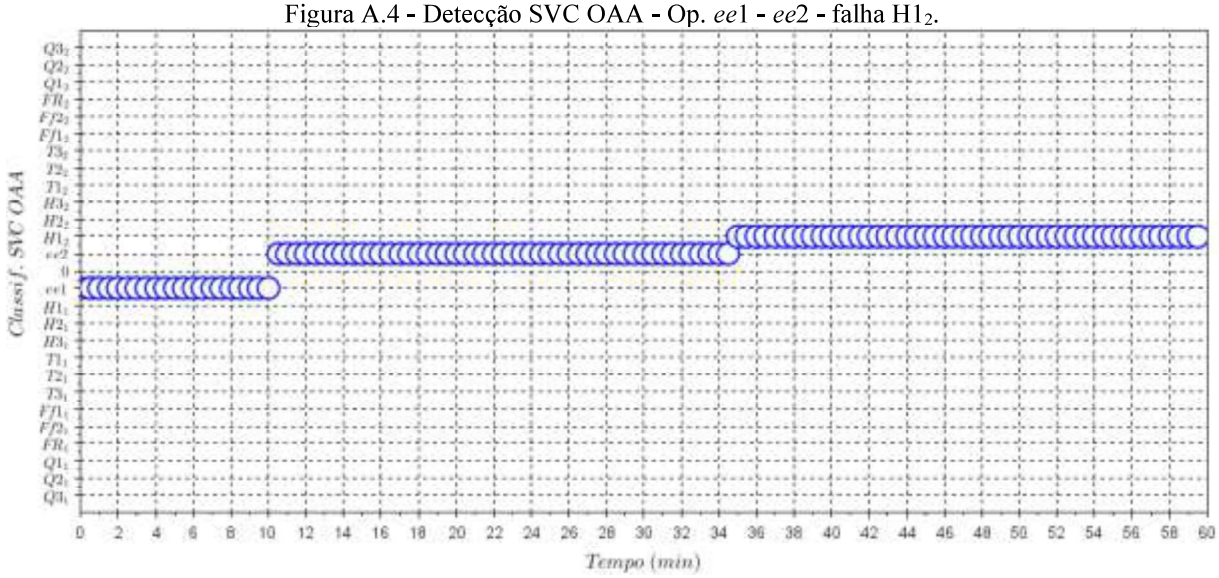


Figura A.5 - Detecção SVC OAA - Op. *ee*2 - *ee*1 - falha $H2_1$.

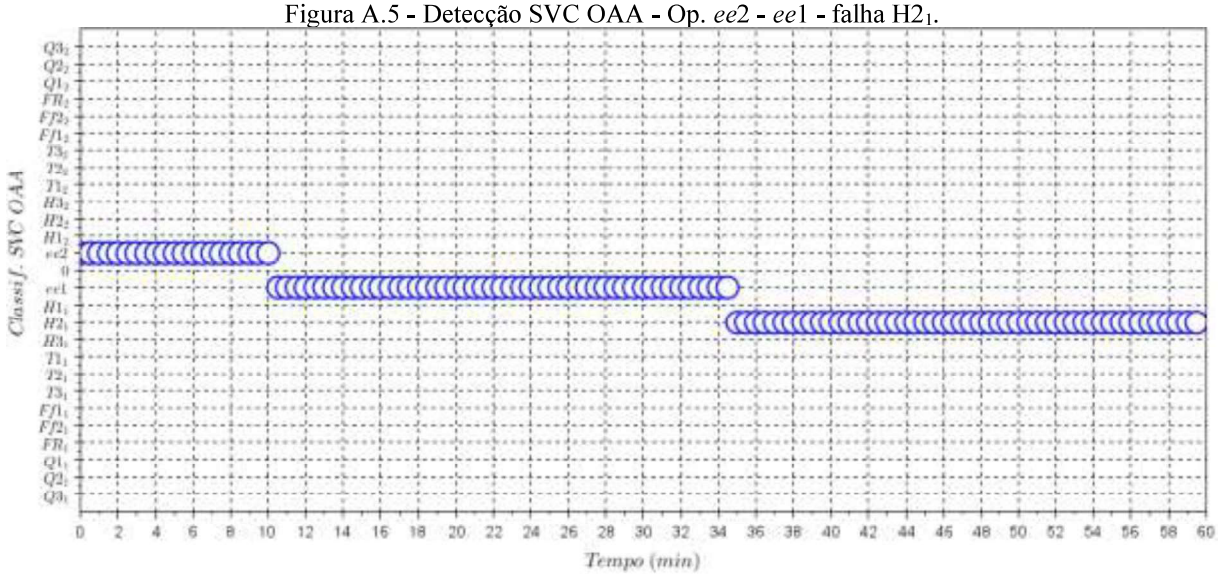


Figura A.6 - Detecção SVC OAA - Op. *ee*1 - *ee*2 - falha $H2_2$.

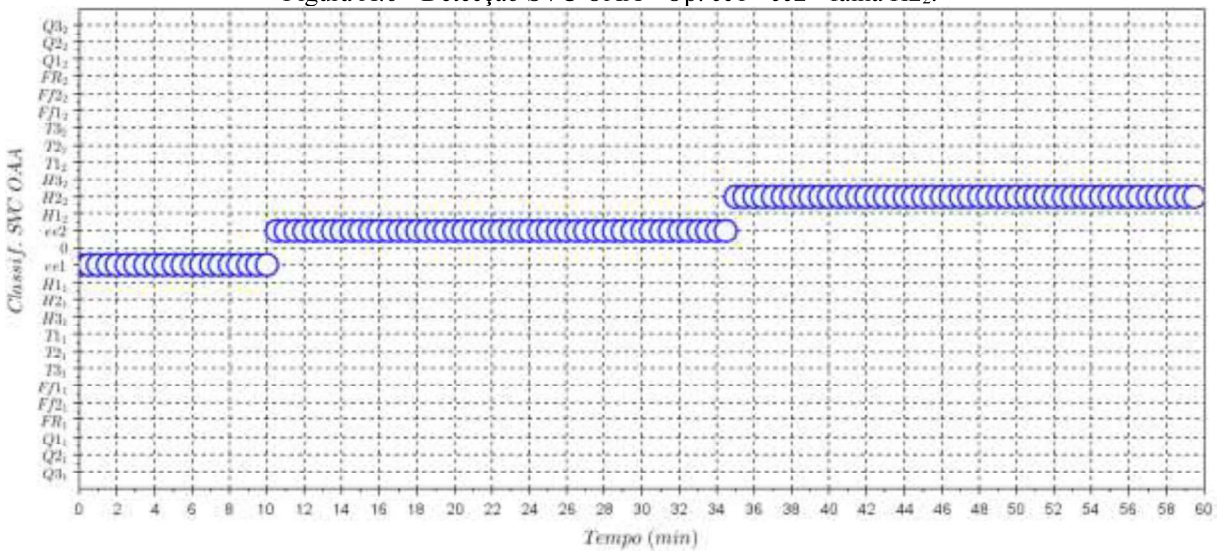


Figura A.7 - Detecção SVC OAA - Op. *ee*2 - *ee*1 - falha $H3_1$.

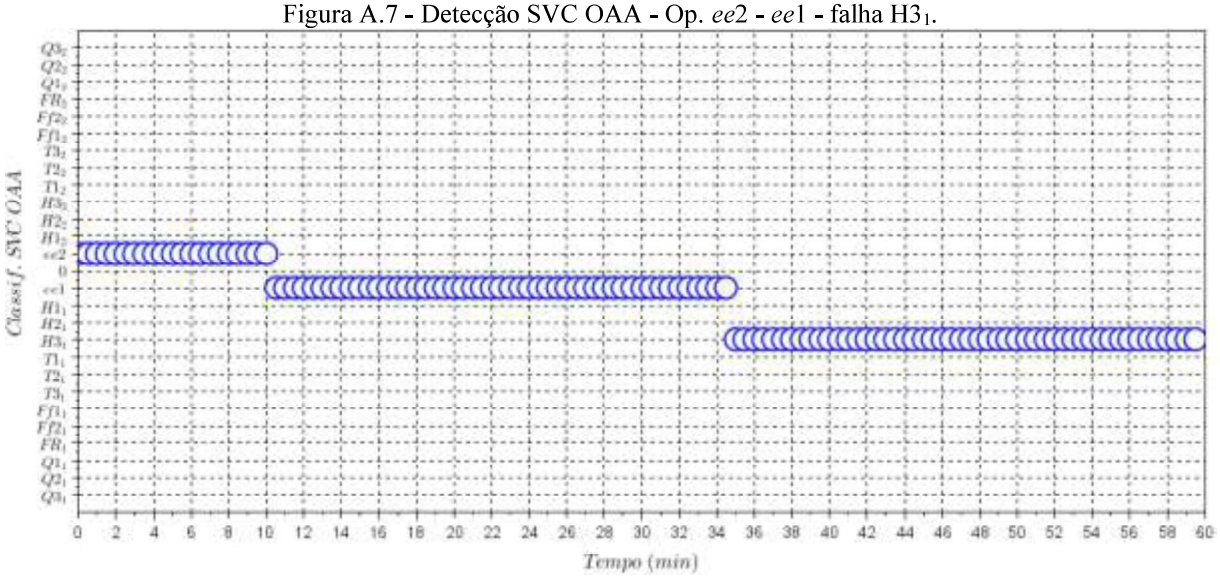


Figura A.8 - Detecção SVC OAA - Op. *ee*1 - *ee*2 - falha $H3_2$.

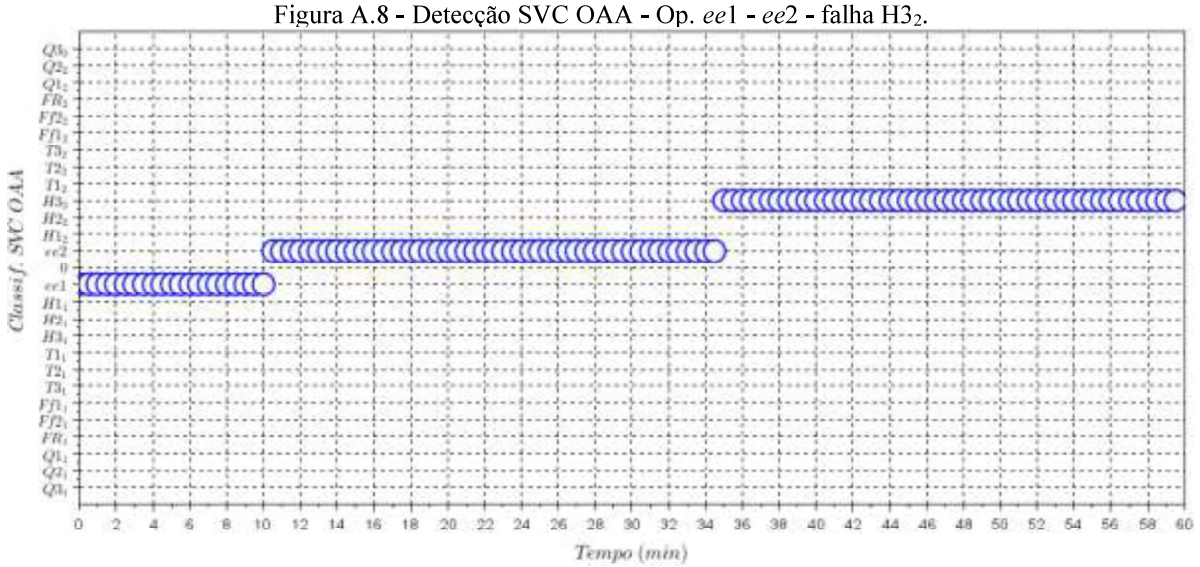


Figura A.9 - Detecção SVC OAA - Op. *ee*2 - *ee*1 - falha $T1_1$.

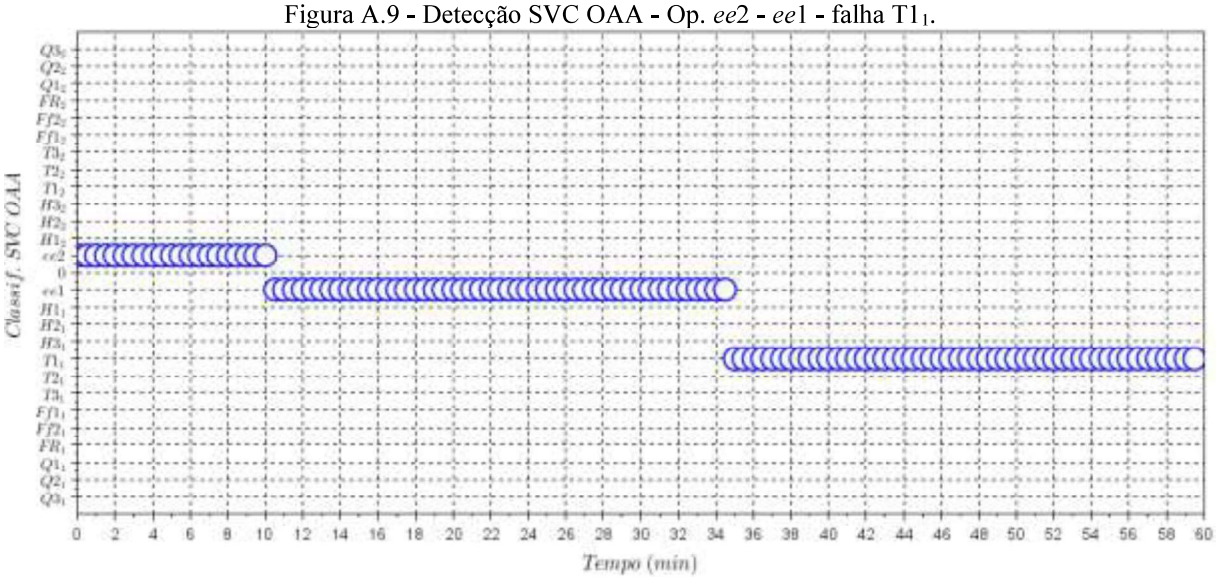


Figura A.10 - Detecção SVC OAA - Op. *ee*1 - *ee*2 - falha $T1_2$.

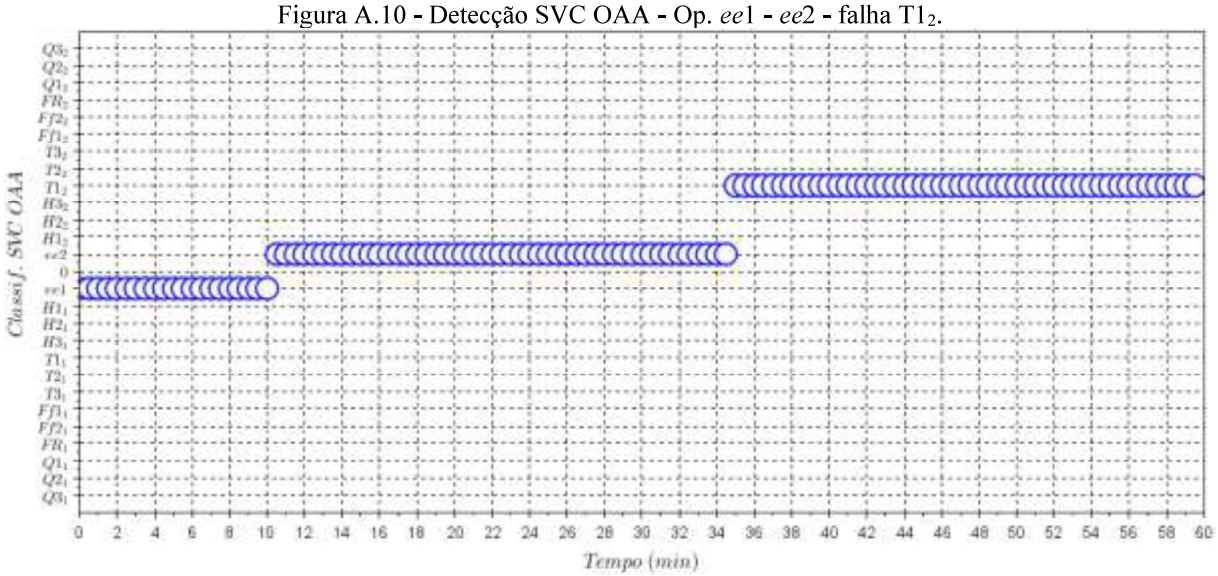


Figura A.11 - Detecção SVC OAA - Op. *ee*2 - *ee*1 - falha $T2_1$.

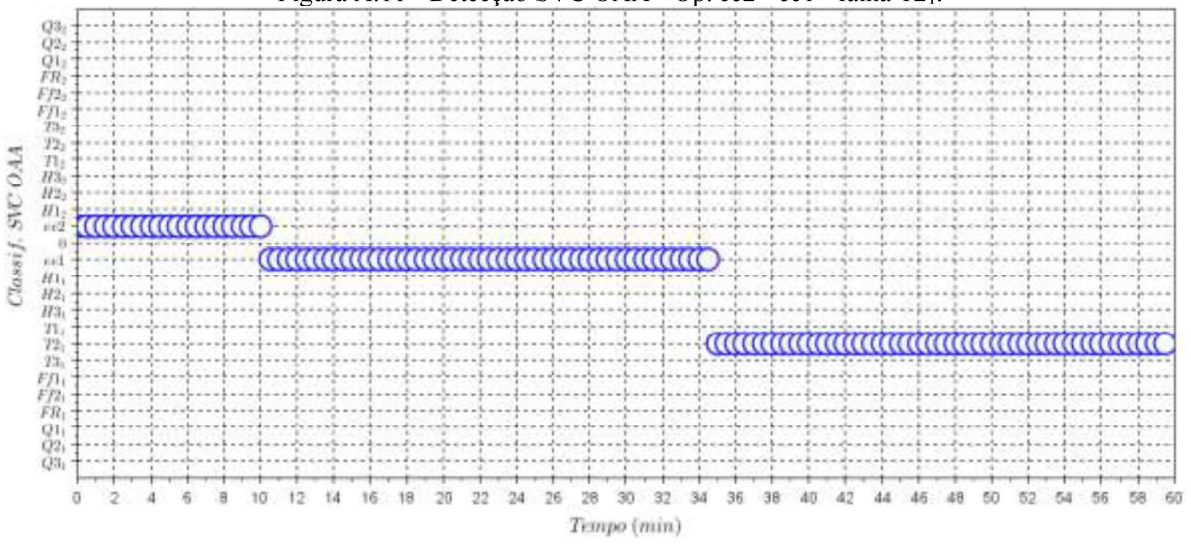


Figura A.12 - Detecção SVC OAA - Op. *ee*1 - *ee*2 - falha $T2_2$.

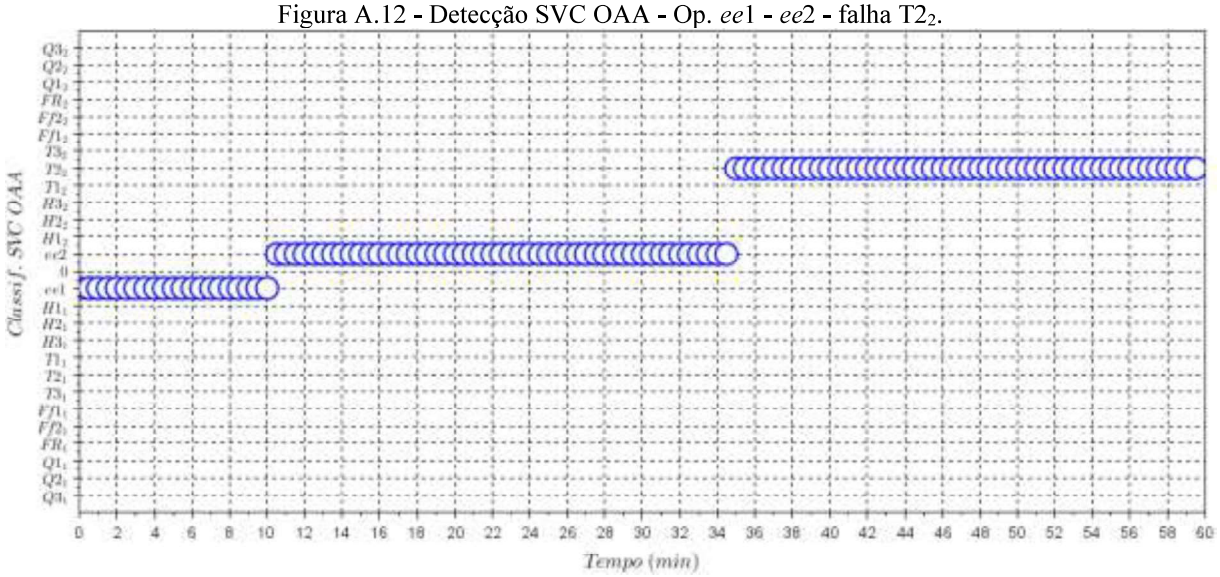


Figura A.13 - Detecção SVC OAA - Op. *ee*2 - *ee*1 - falha $T3_1$.

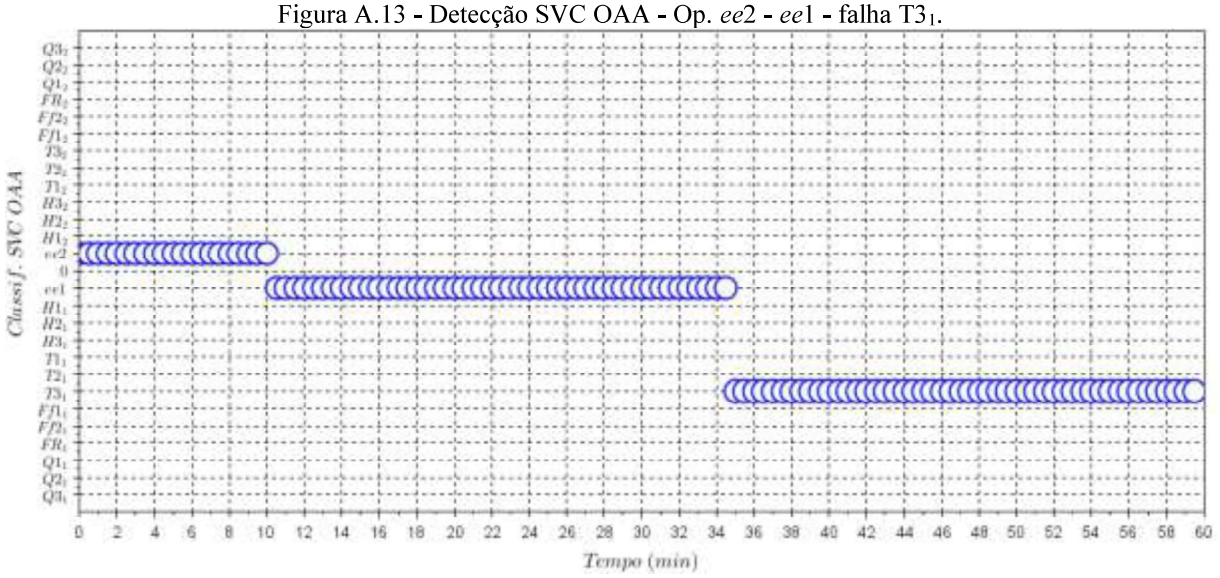


Figura A.14 - Detecção SVC OAA - Op. *ee*1 - *ee*2 - falha $T3_2$.

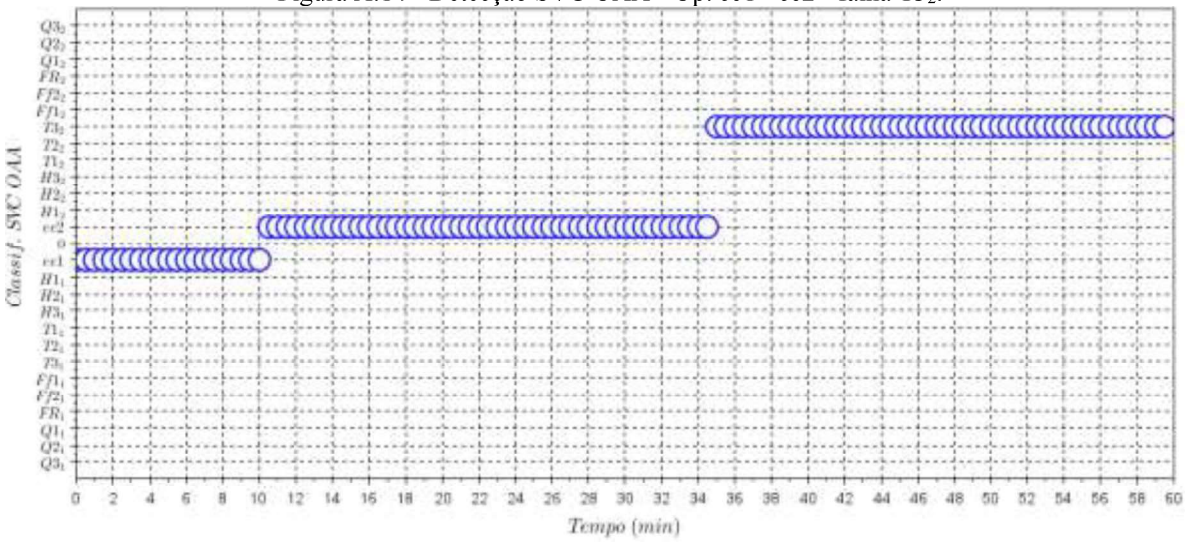


Figura A.15 - Detecção SVC OAA - Op. *ee*2 - *ee*1 - falha $Ff1_1$.

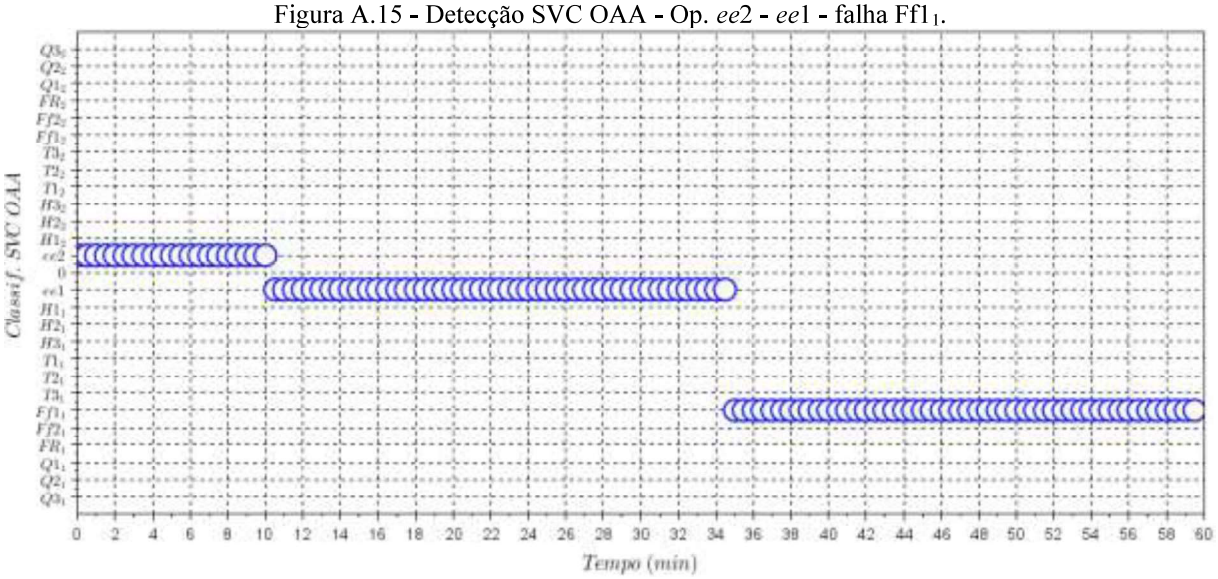


Figura A.16 - Detecção SVC OAA - Op. *ee*1 - *ee*2 - falha $Ff1_2$.

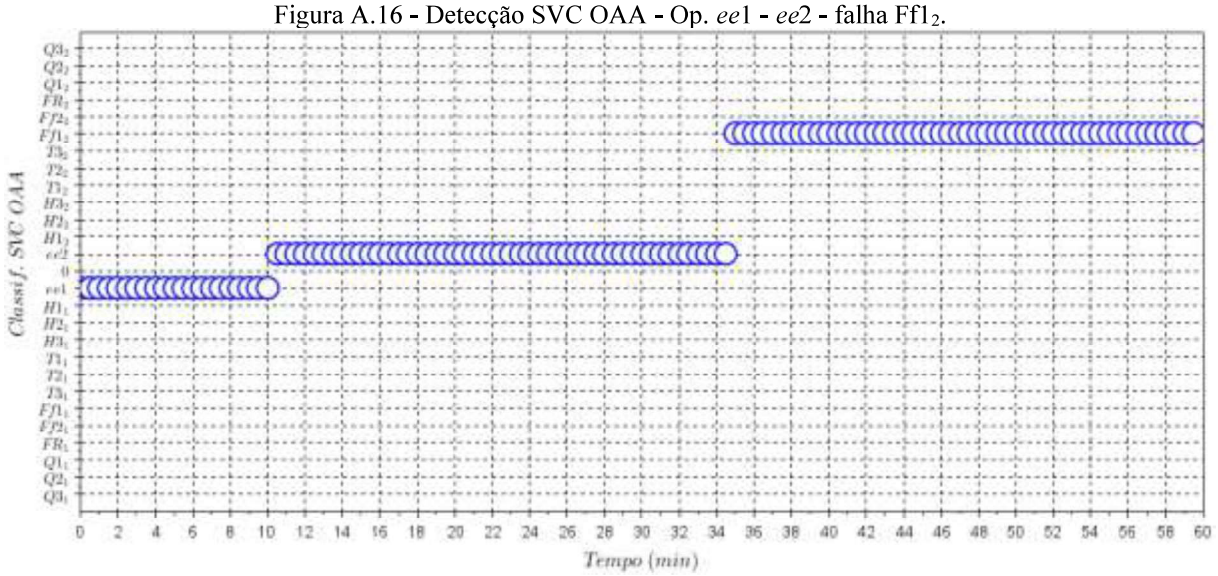


Figura A.17 - Detecção SVC OAA - Op. *ee*2 - *ee*1 - falha $Ff2_1$.

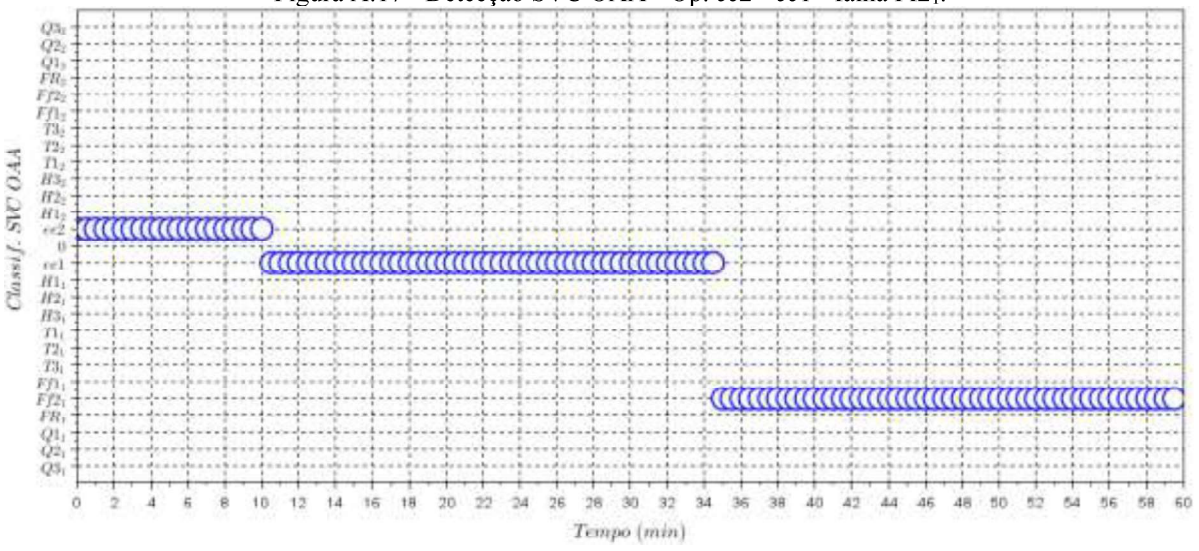


Figura A.18 - Detecção SVC OAA - Op. *ee*1 - *ee*2 - falha $Ff2_2$.

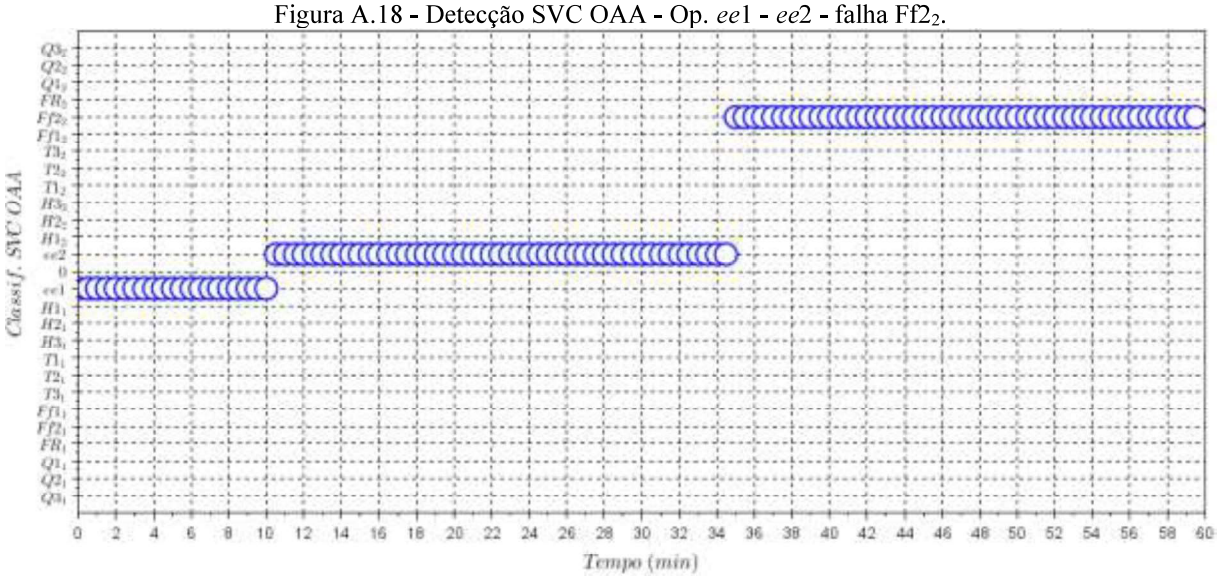


Figura A.19 - Detecção SVC OAA - Op. *ee*2 - *ee*1 - falha FR_1.

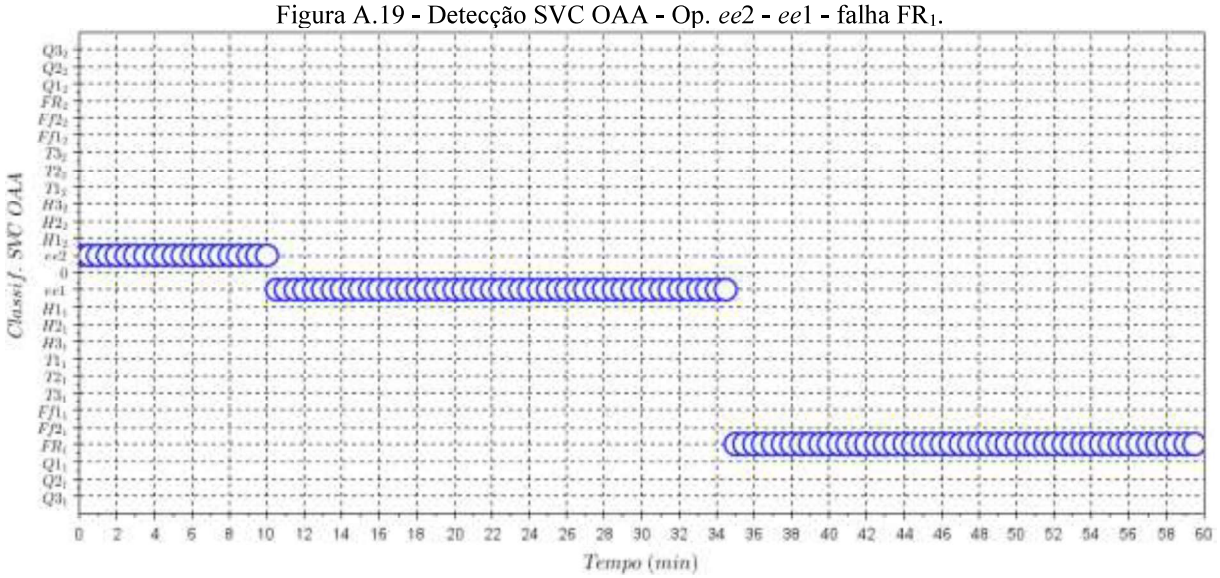


Figura A.20 - Detecção SVC OAA - Op. *ee*1 - *ee*2 - falha FR_2.

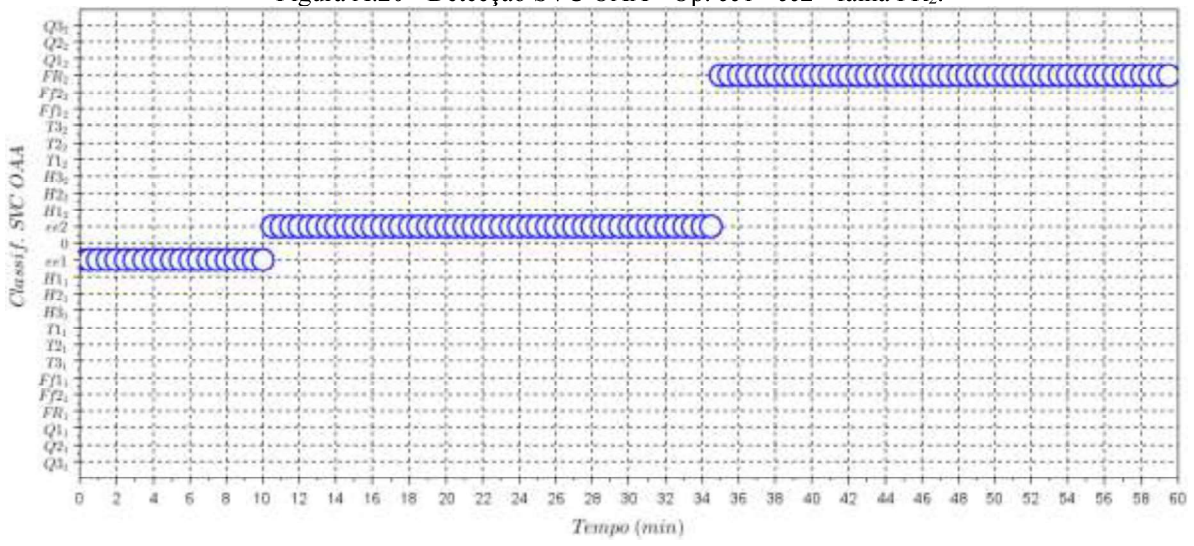


Figura A.21 - Detecção SVC OAA - Op. *ee*2 - *ee*1 - falha $Q1_1$.

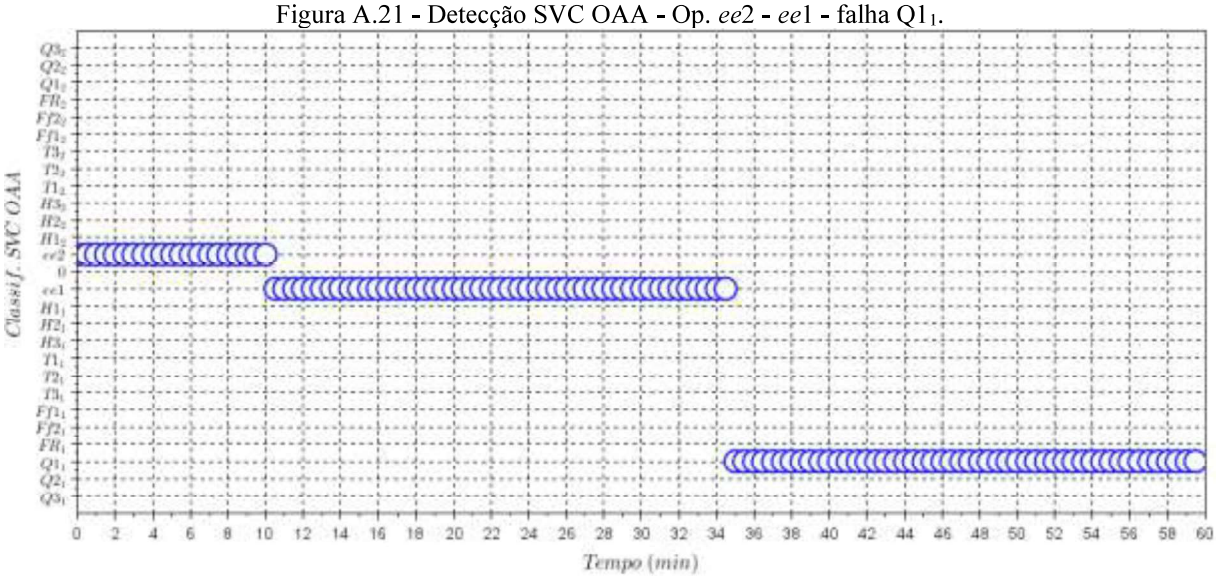


Figura A.22 - Detecção SVC OAA - Op. *ee*1 - *ee*2 - falha $Q1_2$.

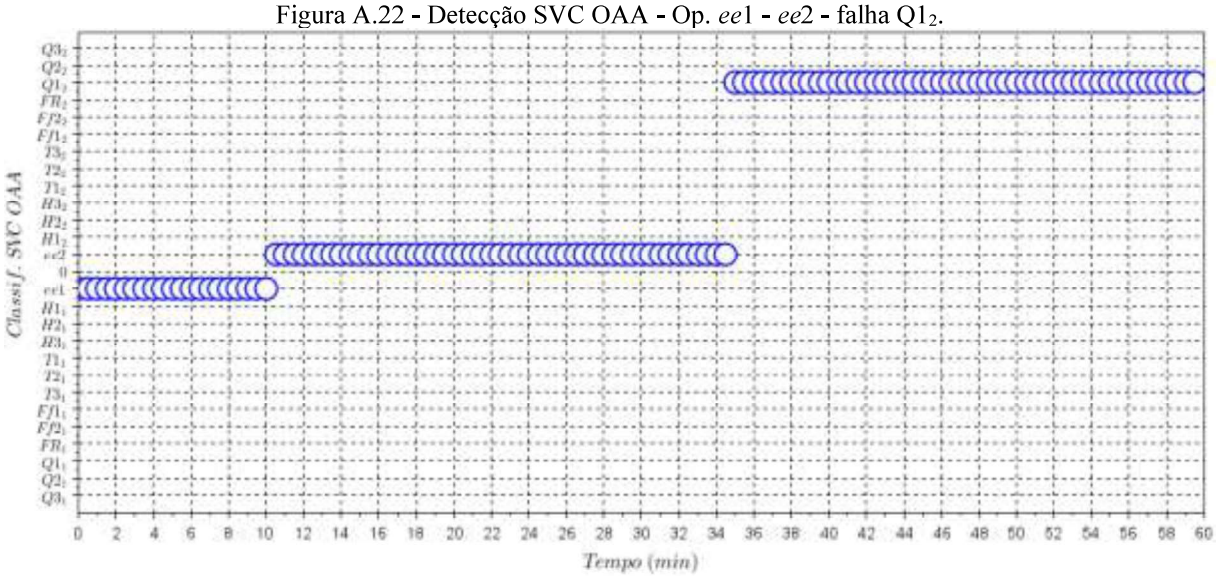


Figura A.23 - Detecção SVC OAA - Op. *ee*2 - *ee*1 - falha $Q2_1$.

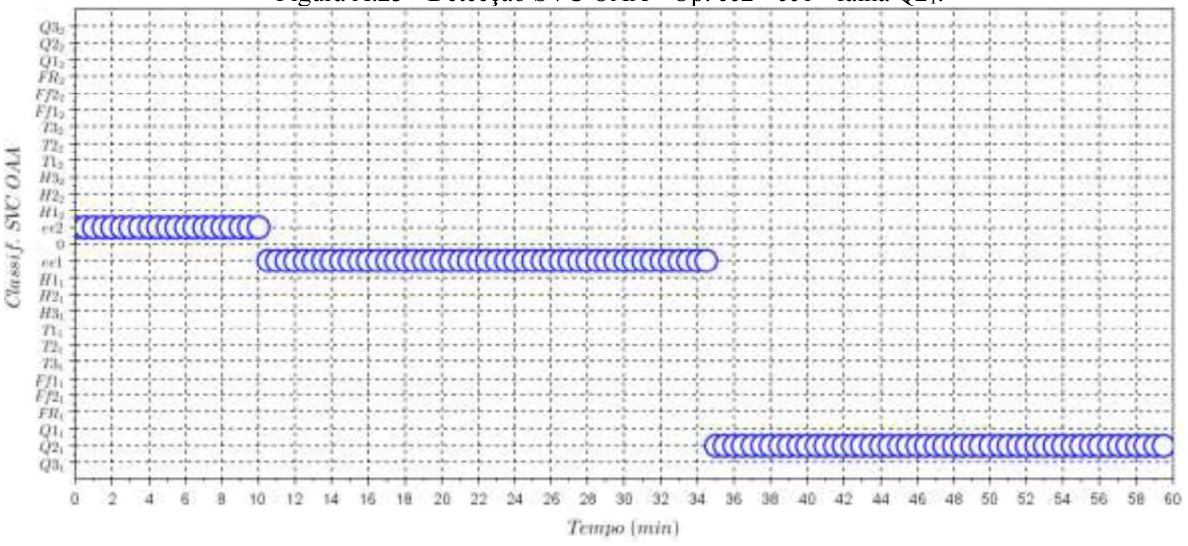


Figura A.24 - Detecção SVC OAA - Op. *ee*1 - *ee*2 - falha $Q2_2$.

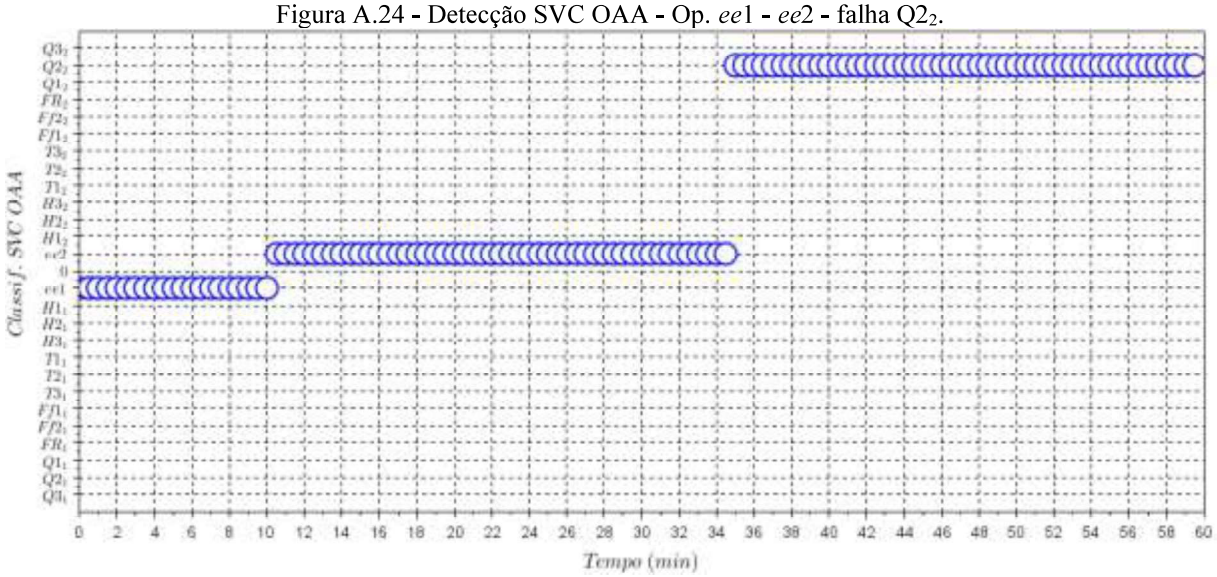


Figura A.25 - Detecção SVC OAA - Op. *ee*2 - *ee*1 - falha $Q3_1$.

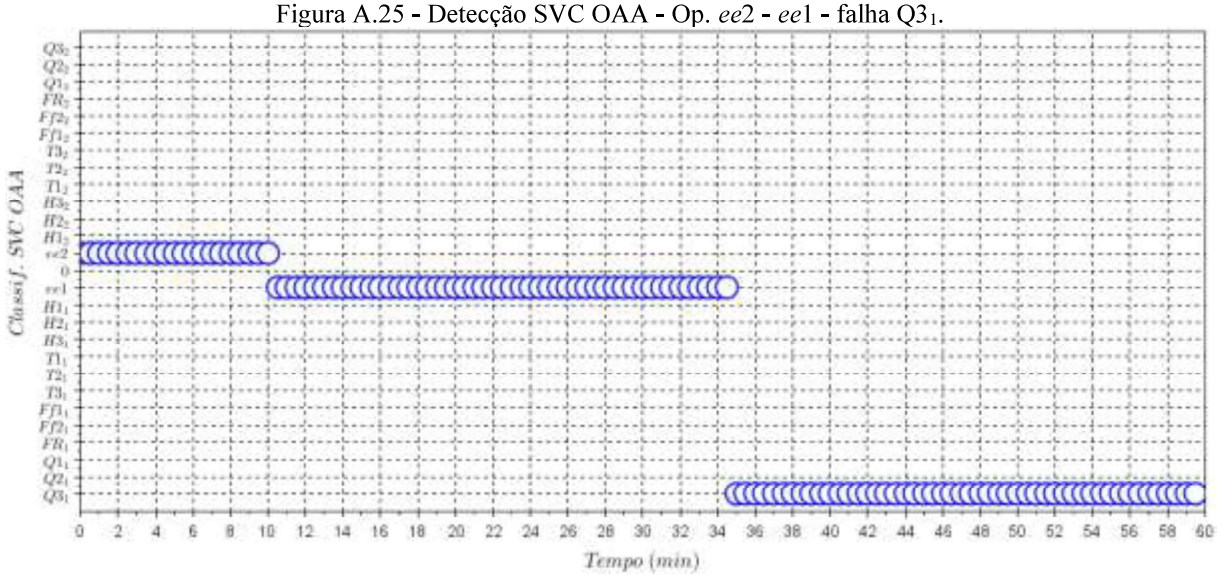
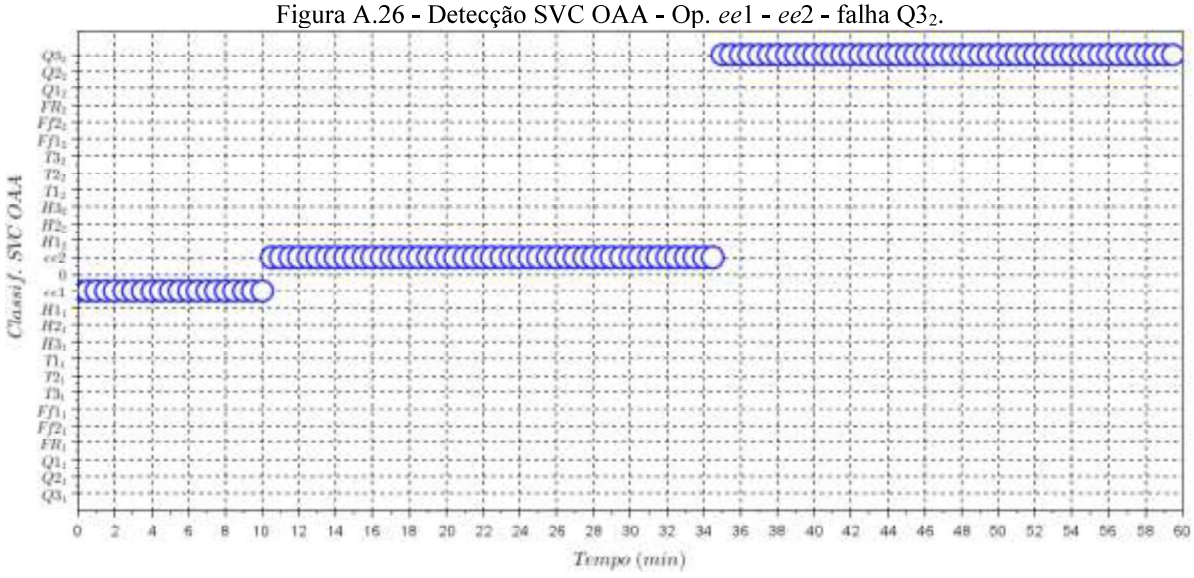


Figura A.26 - Detecção SVC OAA - Op. *ee*1 - *ee*2 - falha $Q3_2$.



A.2. Resultados SVC OAO

Foram estabelecidas 24 falhas, sendo duas falhas para cada variável, aplicadas em cada um dos dois estados estacionários estipulados, de acordo com o Capítulo 3. Além das 24 falhas, foram utilizados dados de ambos os estados estacionários, totalizando 26 situações passíveis de detecção. A metodologia de detecção de falhas SVC *one against one* utiliza o conjunto de dados e rótulos estabelecidos para os sinais de treinamento de forma a criar hiperplanos ótimos que separam cada diferente classe (falhas e operações normais) de cada uma das outras. A metodologia *one against one* gera modelos que separam uma classe de cada uma das outras classes existentes, resultando assim em $n(n\text{-}1)/2$ modelos diferentes, sendo n o número de classes.

Para todas as situações passíveis de detecção, foram treinados sinais em que era aplicada a falha (ou outro estado estacionário), nos instantes 10 minutos e 35 minutos, em um total de 60 minutos de operação, para cada sinal treinado. Os dados então foram separados em dados de treinamento, de validação e de teste. Foi utilizado uma proporção de 2 para 1 entre os dados de treinamento e validação. A cada dois instantes utilizados para treinamento, o terceiro instante era separado para validação. Para os dados de teste, foram simulados outro conjunto de dados cujos instantes de aplicação de falha e modificação de estado estacionário eram diferentes dos dados treinados e validados.

Nas figuras a seguir, apresentam-se os resultados obtidos para o sistema de detecção, para todas as situações apresentadas na Tabela A.1.

Figura A.27 - Detecção SVC OAO - Op. *ee*2 - *ee*1 - *ee*2.

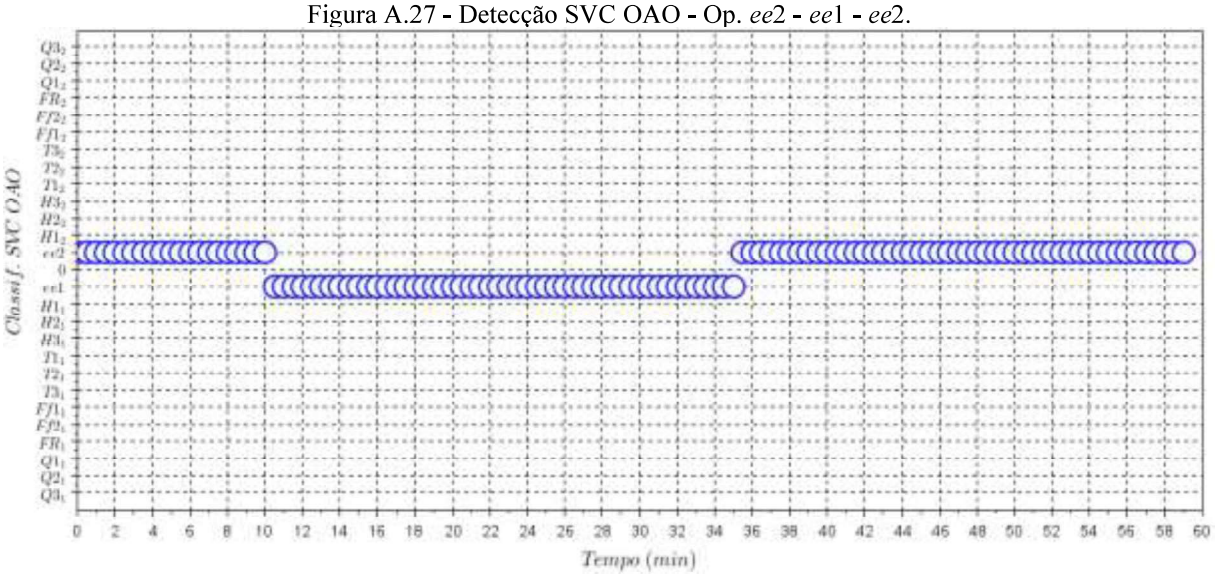


Figura A.28 - Detecção SVC OAO - Op. *ee*1 - *ee*2 - *ee*1.

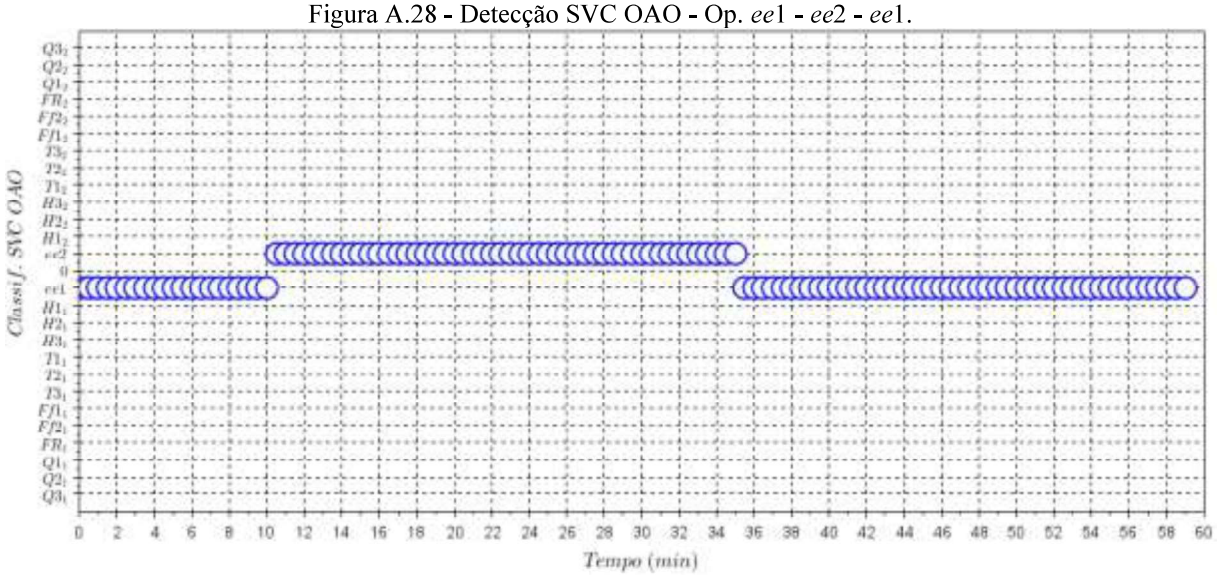


Figura A.29 - Detecção SVC OAO - Op. *ee*2 - *ee*1 - falha $H1_1$.

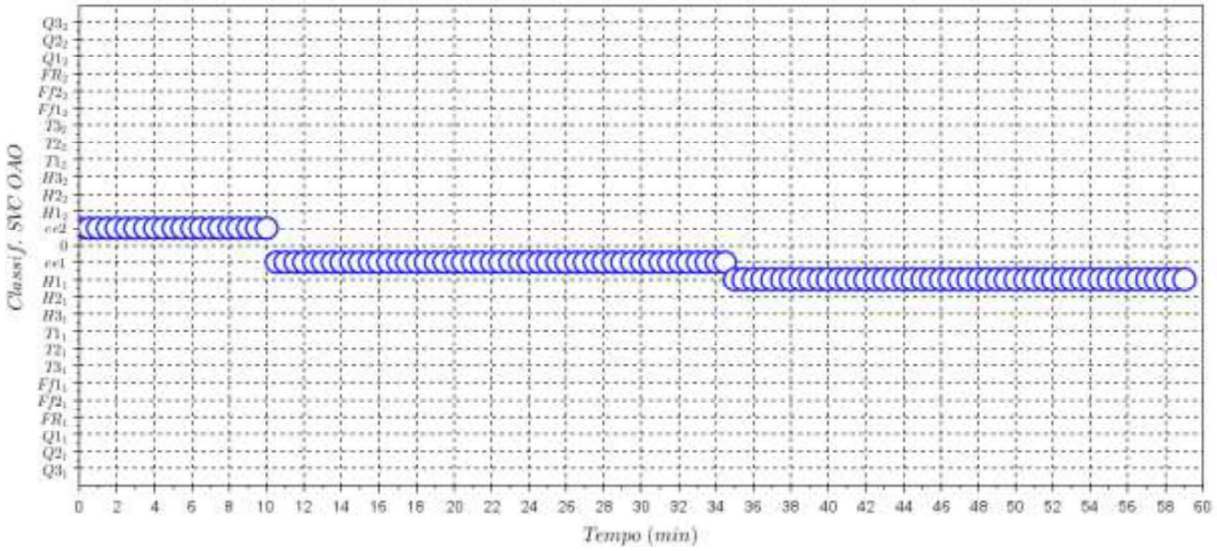


Figura A.30 - Detecção SVC OAO - Op. *ee*1 - *ee*2 - falha $H1_2$.

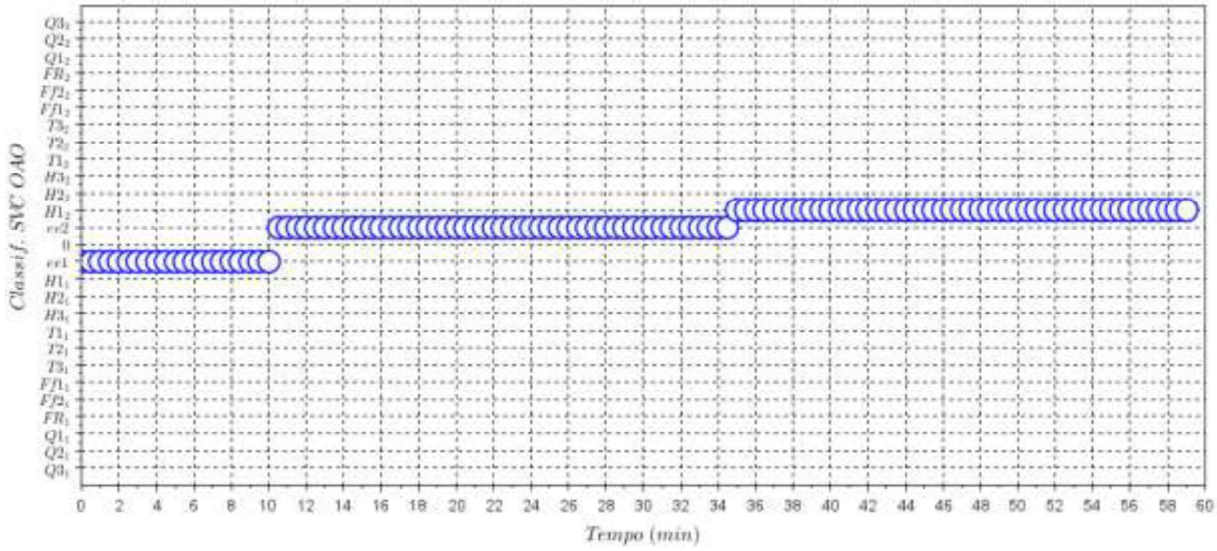


Figura A.31 - Detecção SVC OAO - Op. *ee*2 - *ee*1 - falha $H2_1$.

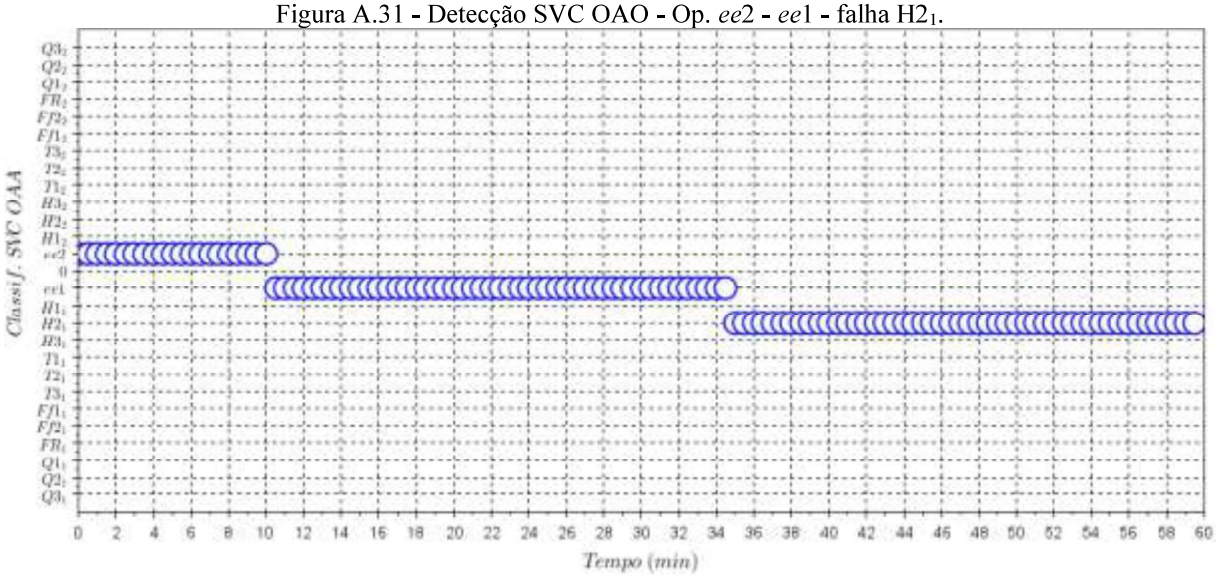


Figura A.32 - Detecção SVC OAO - Op. *ee*1 - *ee*2 - falha $H2_2$.

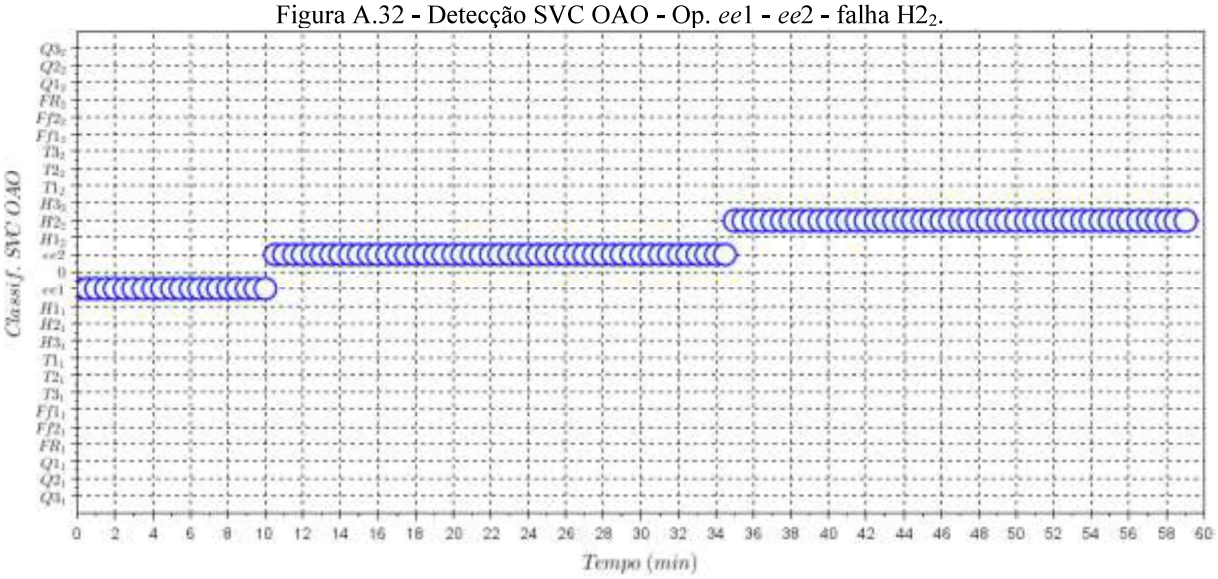


Figura A.33 - Detecção SVC OAO - Op. *ee*2 - *ee*1 - falha $H3_1$.

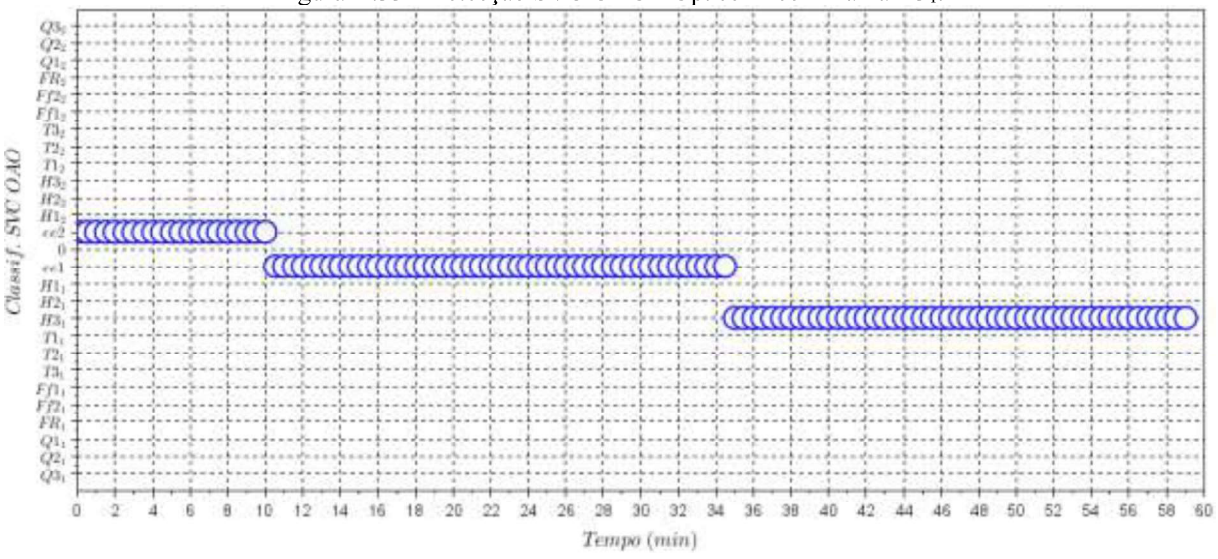


Figura A.34 - Detecção SVC OAO - Op. *ee*1 - *ee*2 - falha $H3_2$.

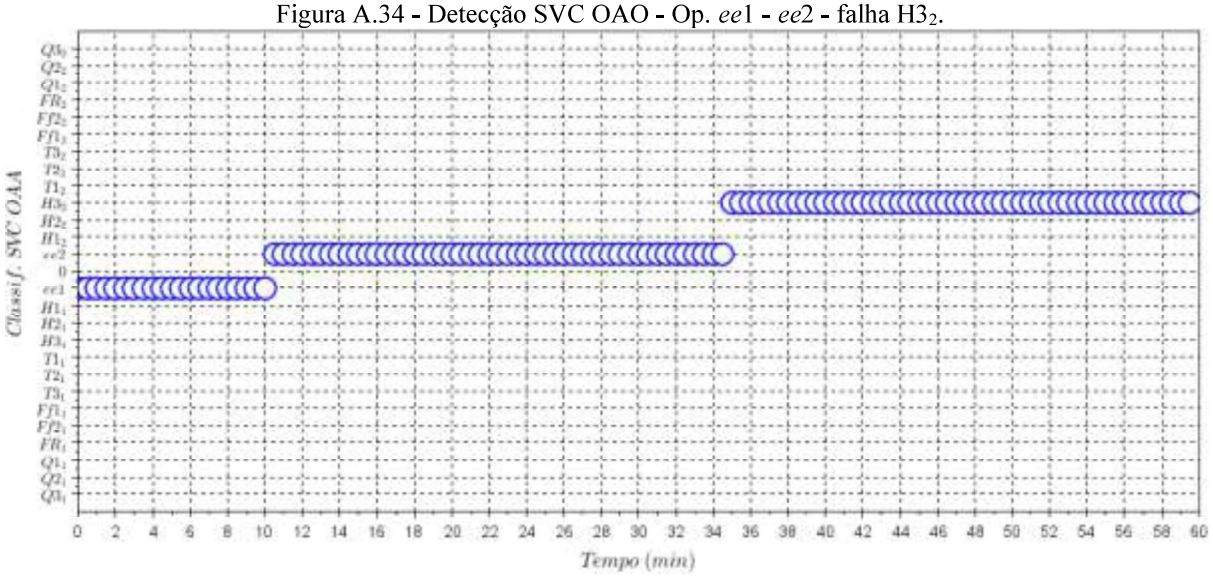


Figura A.35 - Detecção SVC OAO - Op. *ee*2 - *ee*1 - falha $T1_1$.

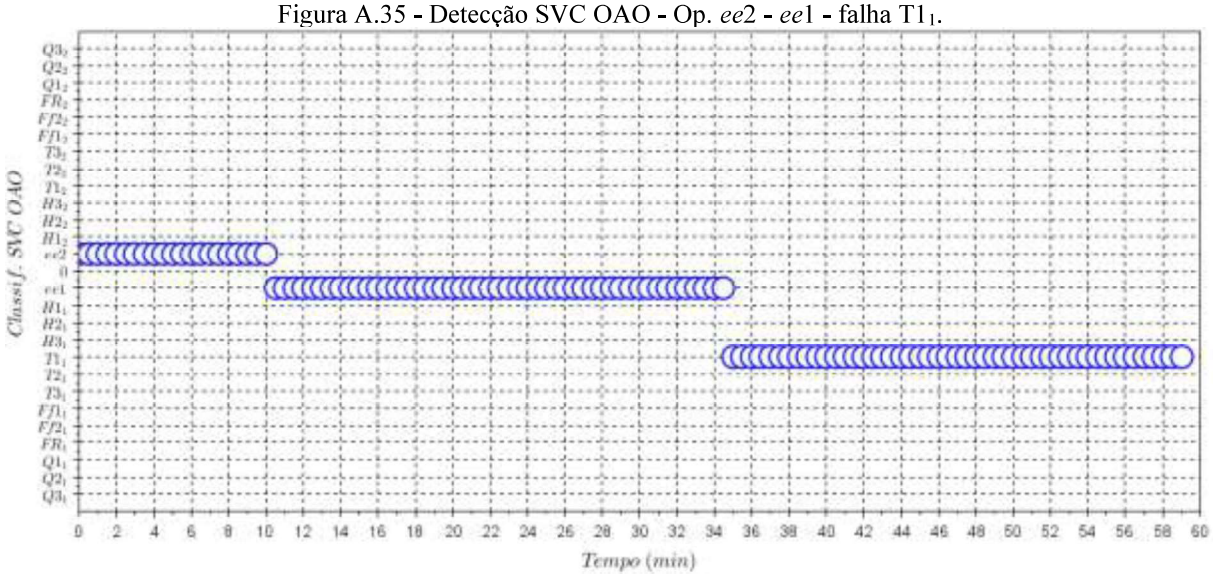


Figura A.36 - Detecção SVC OAO - Op. *ee*1 - *ee*2 - falha $T1_2$.

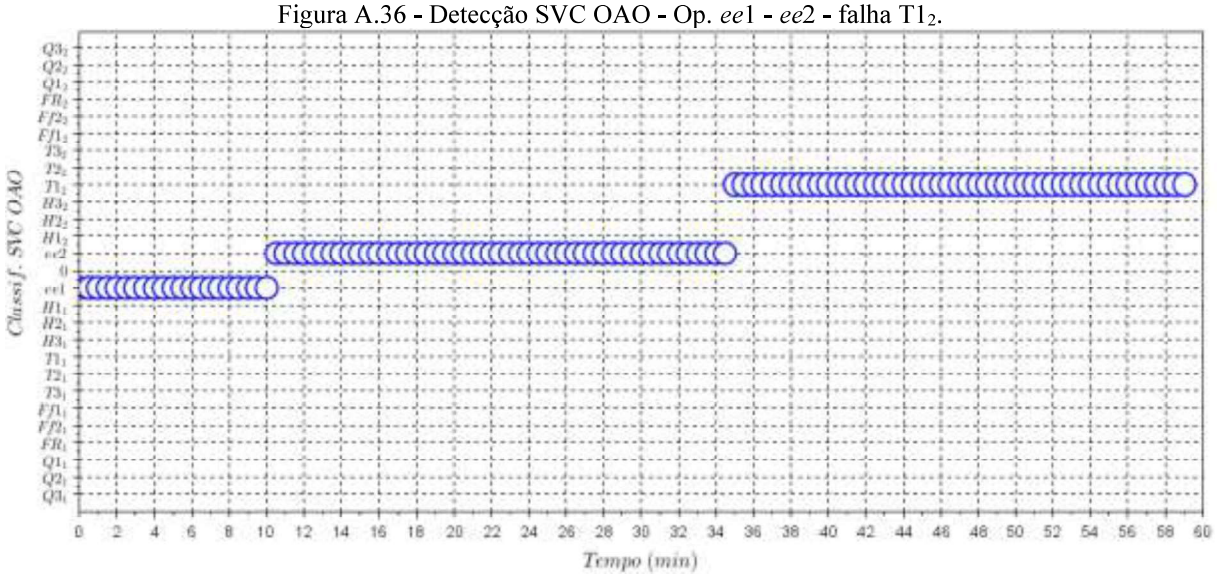


Figura A.37 - Detecção SVC OAO - Op. *ee*2 - *ee*1 - falha $T2_1$.

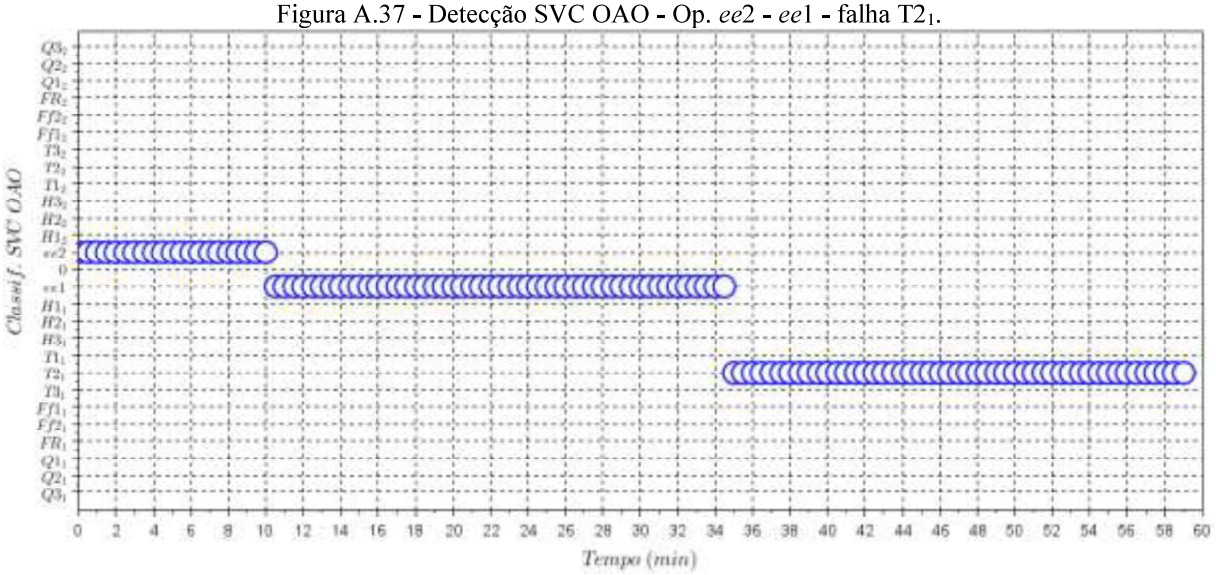


Figura A.38 - Detecção SVC OAO - Op. *ee*1 - *ee*2 - falha $T2_2$.

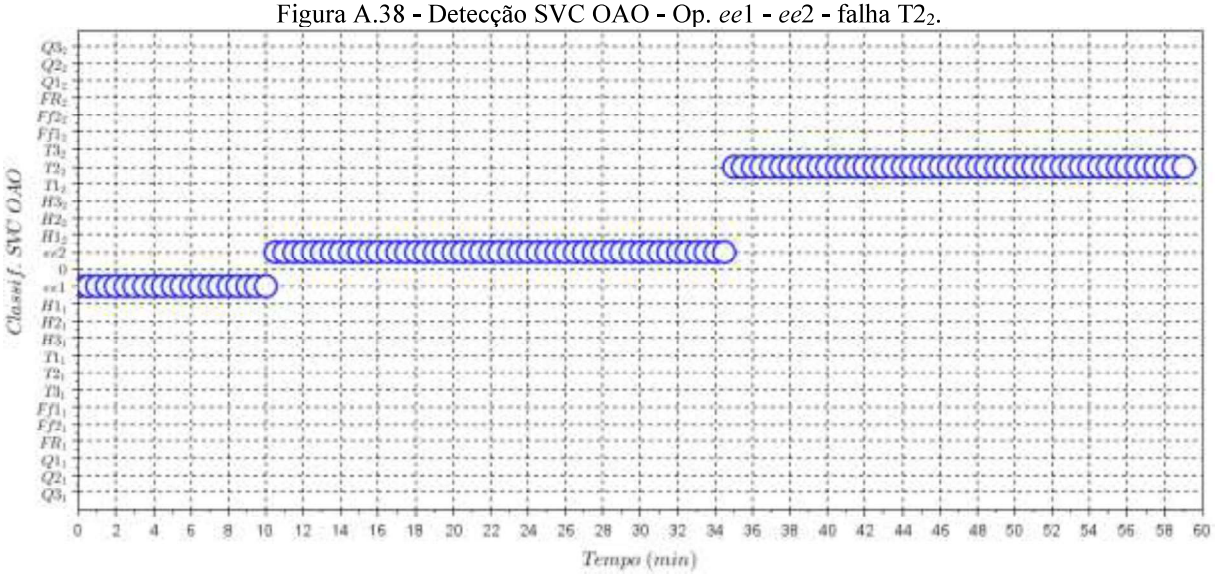


Figura A.39 - Detecção SVC OAO - Op. *ee*2 - *ee*1 - falha $T3_1$.

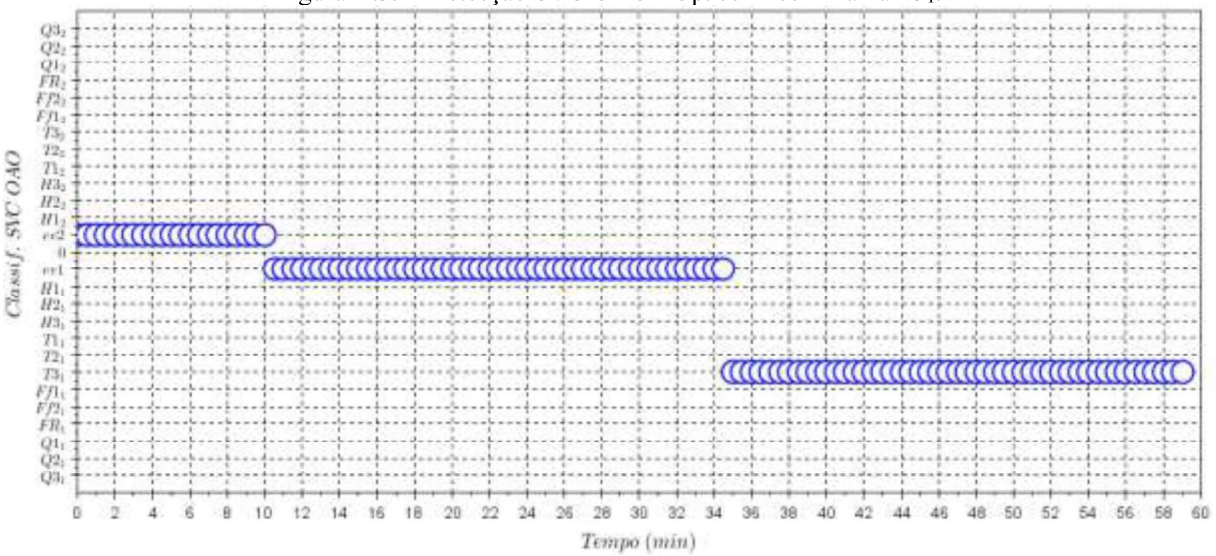


Figura A.40 - Detecção SVC OAO - Op. *ee*1 - *ee*2 - falha $T3_2$.

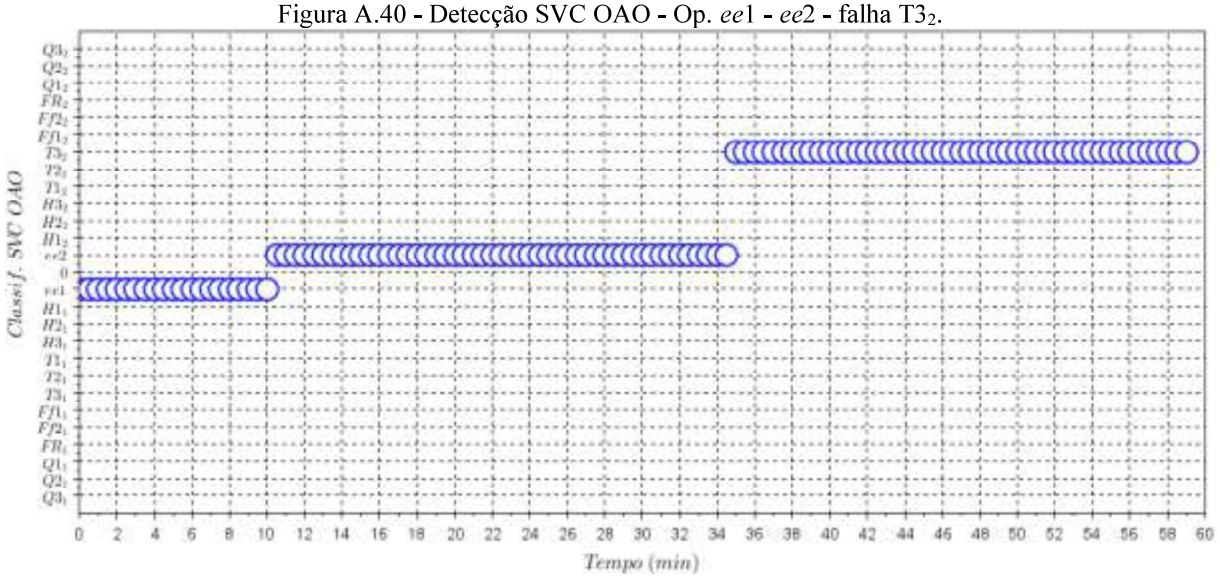


Figura A.41 - Detecção SVC OAO - Op. *ee*2 - *ee*1 - falha $Ff1_1$.

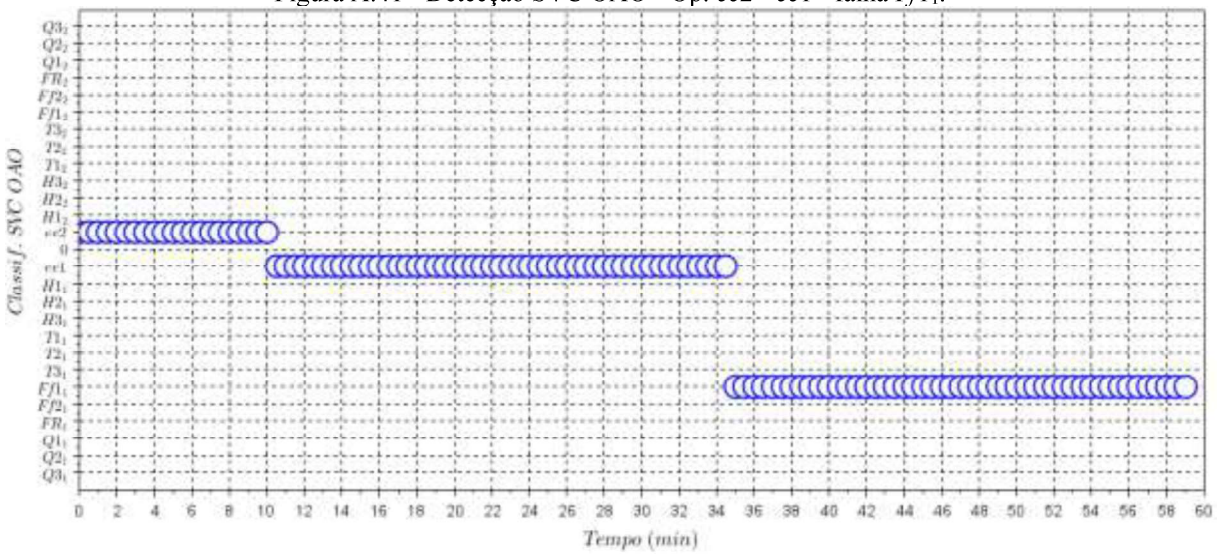


Figura A.42 - Detecção SVC OAO - Op. *ee*1 - *ee*2 - falha $Ff1_2$.

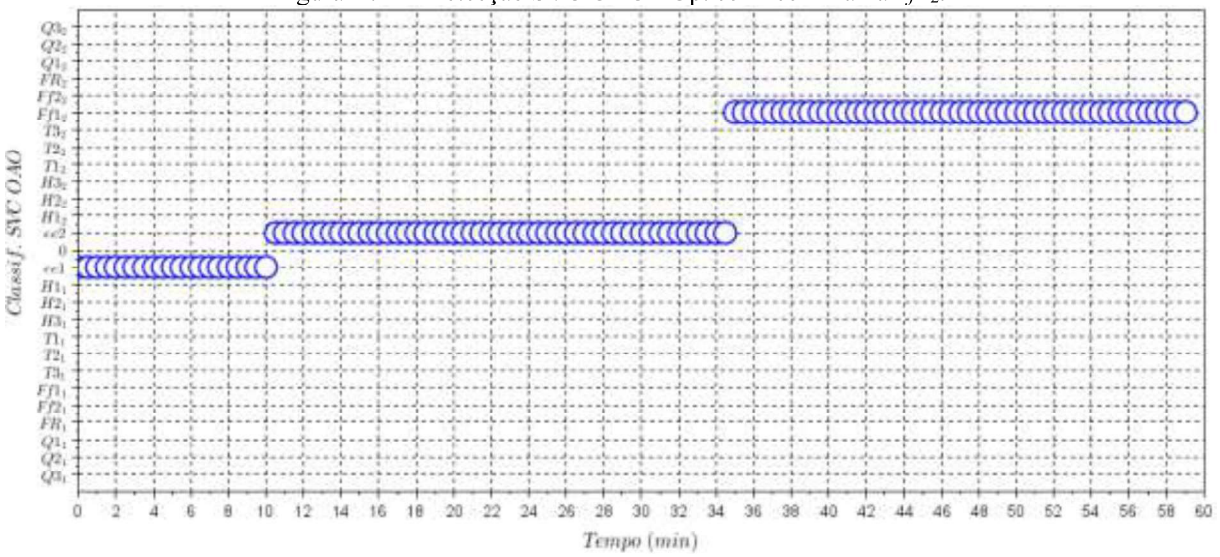


Figura A.43 - Detecção SVC OAO - Op. *ee*2 - *ee*1 - falha F$f2_1$.

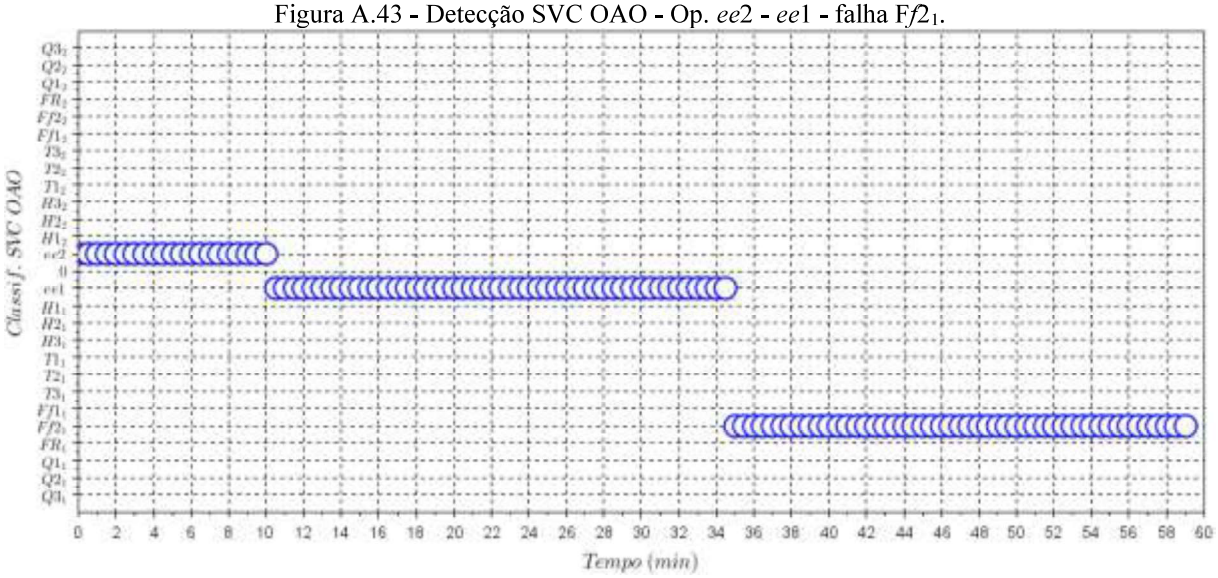


Figura A.44 - Detecção SVC OAO - Op. *ee*1 - *ee*2 - falha F$f2_2$.

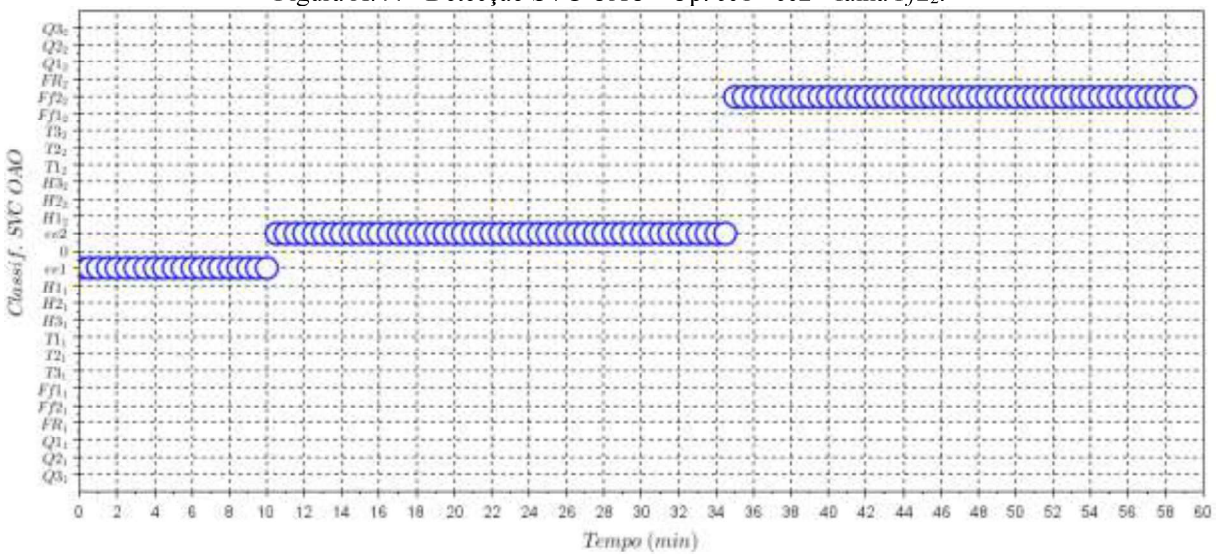


Figura A.45 - Detecção SVC OAO - Op. *ee*2 - *ee*1 - falha F_{R1}.

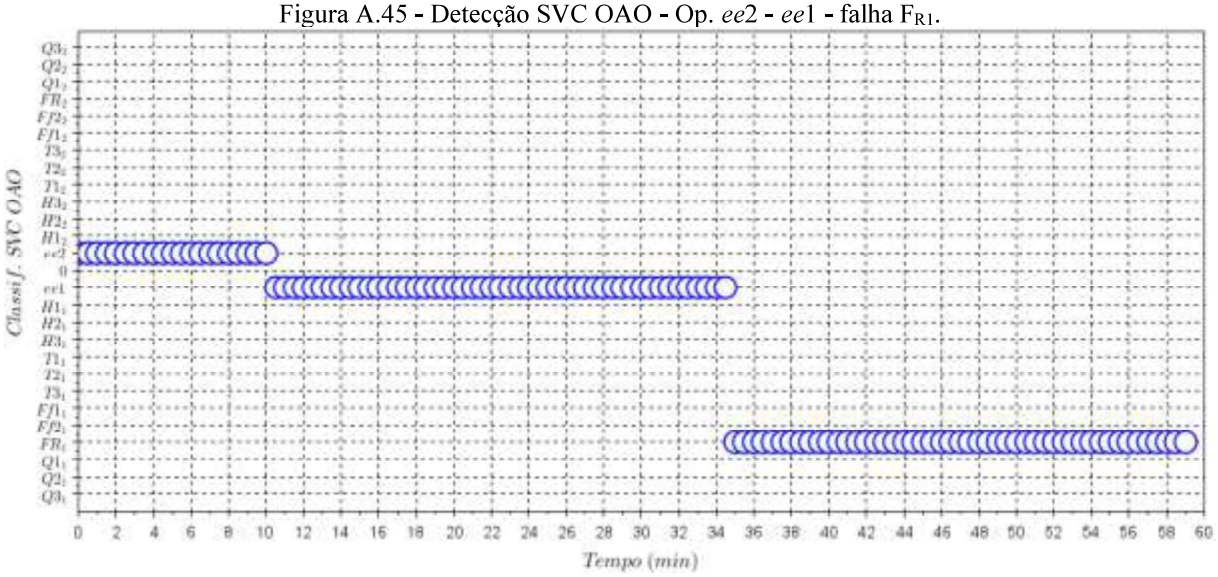


Figura A.46 - Detecção SVC OAO - Op. *ee*1 - *ee*2 - falha F_{R2}.

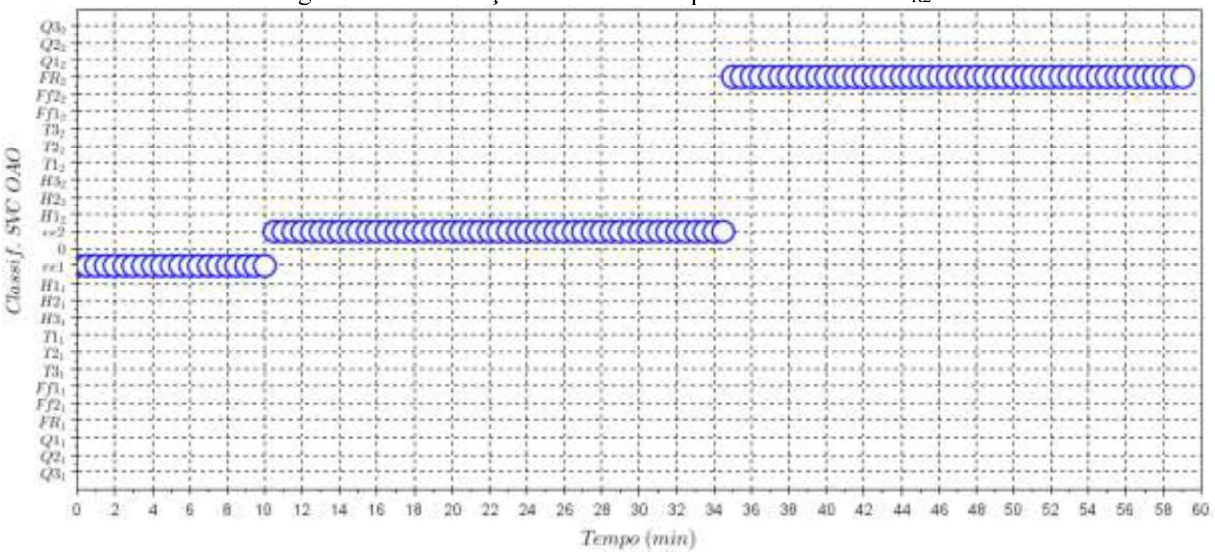


Figura A.47 - Detecção SVC OAO - Op. *ee*2 - *ee*1 - falha $Q1_1$.

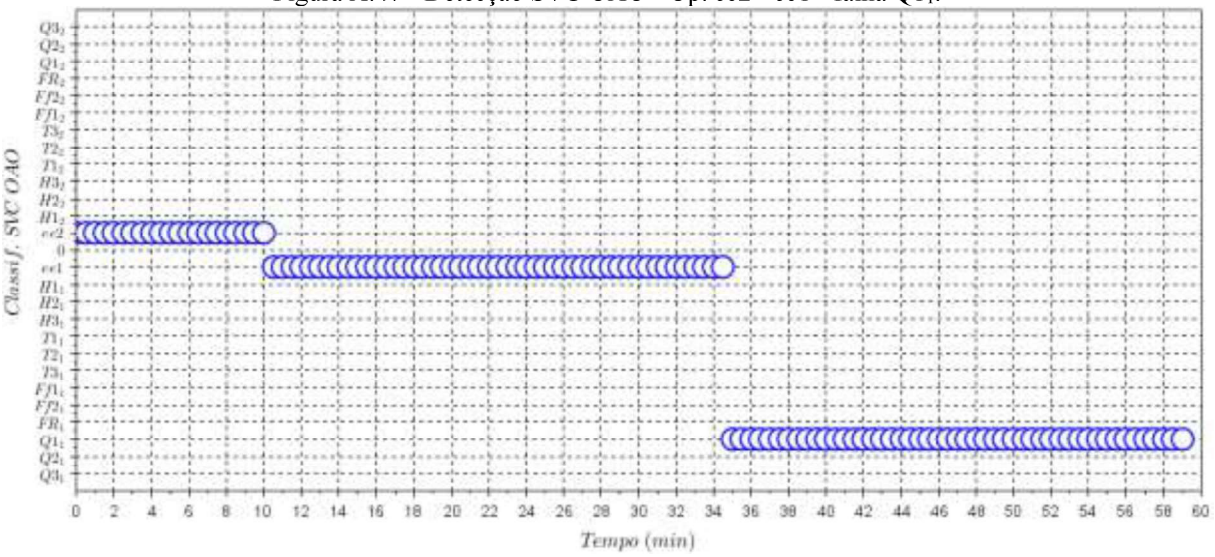


Figura A.48 - Detecção SVC OAO - Op. *ee*1 - *ee*2 - falha $Q1_2$.

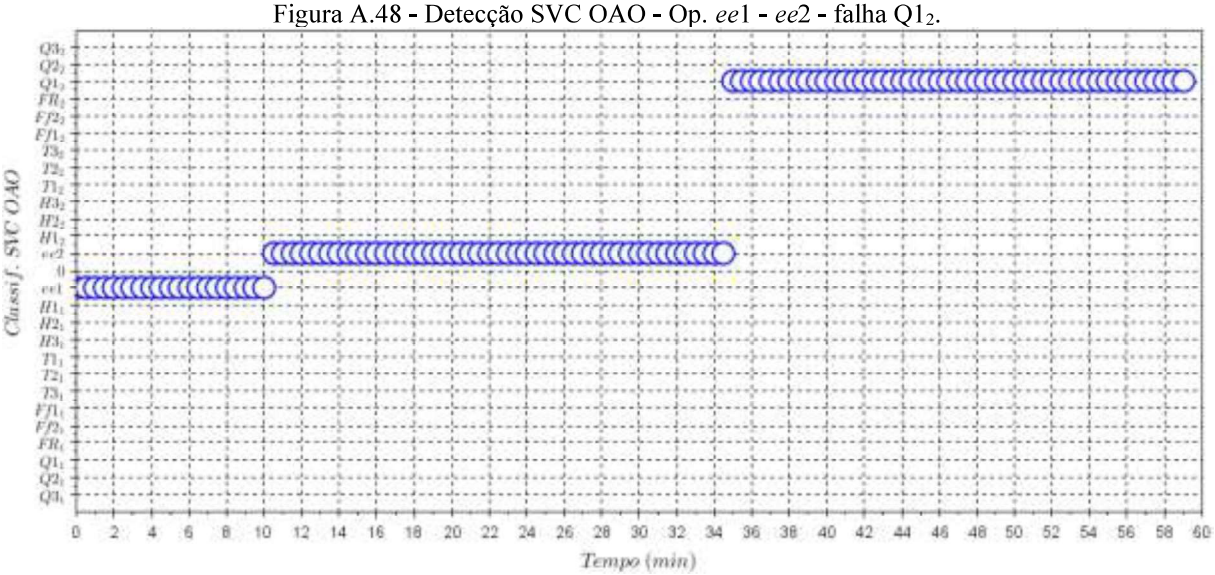


Figura A.49 - Detecção SVC OAO - Op. *ee*2 - *ee*1 - falha $Q2_1$.

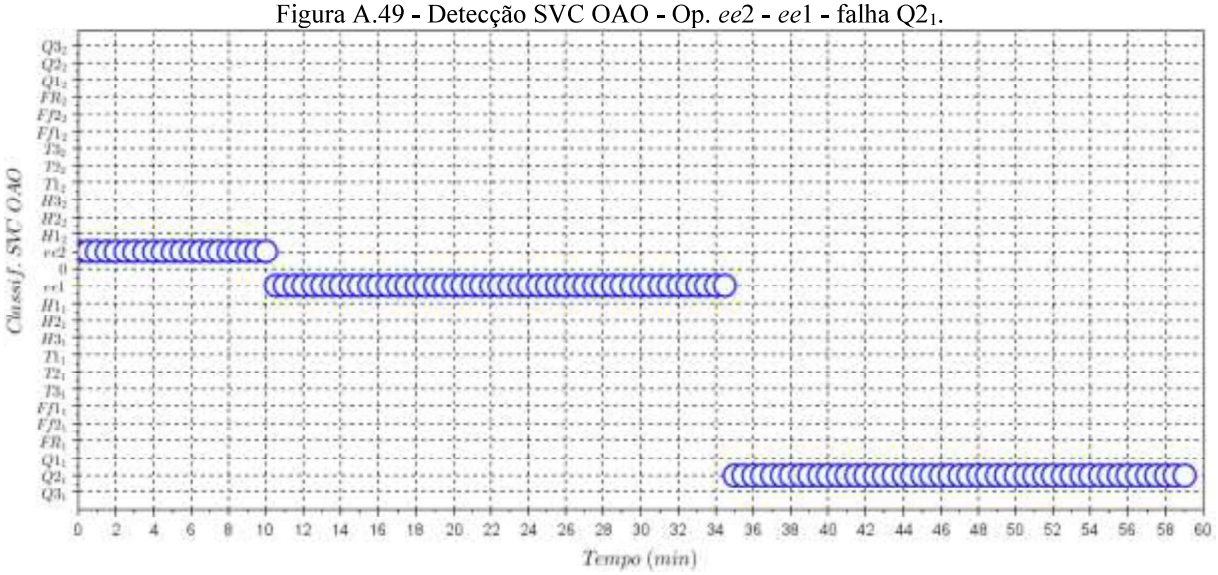


Figura A.50 - Detecção SVC OAO - Op. *ee*1 - *ee*2 - falha $Q2_2$.

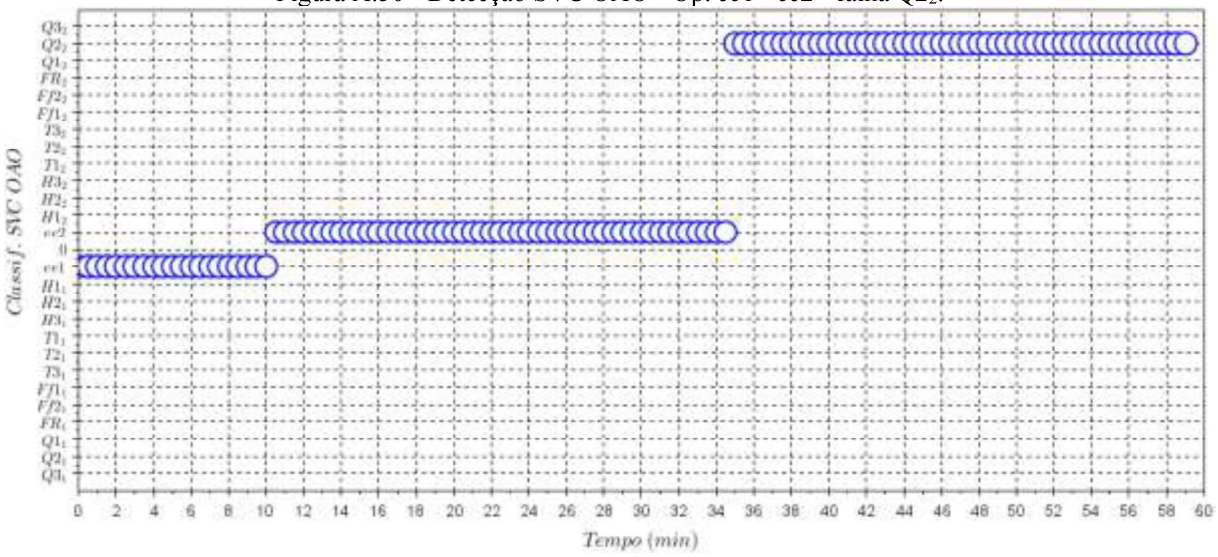


Figura A.51 - Detecção SVC OAO - Op. *ee*2 - *ee*1 - falha $Q3_1$.

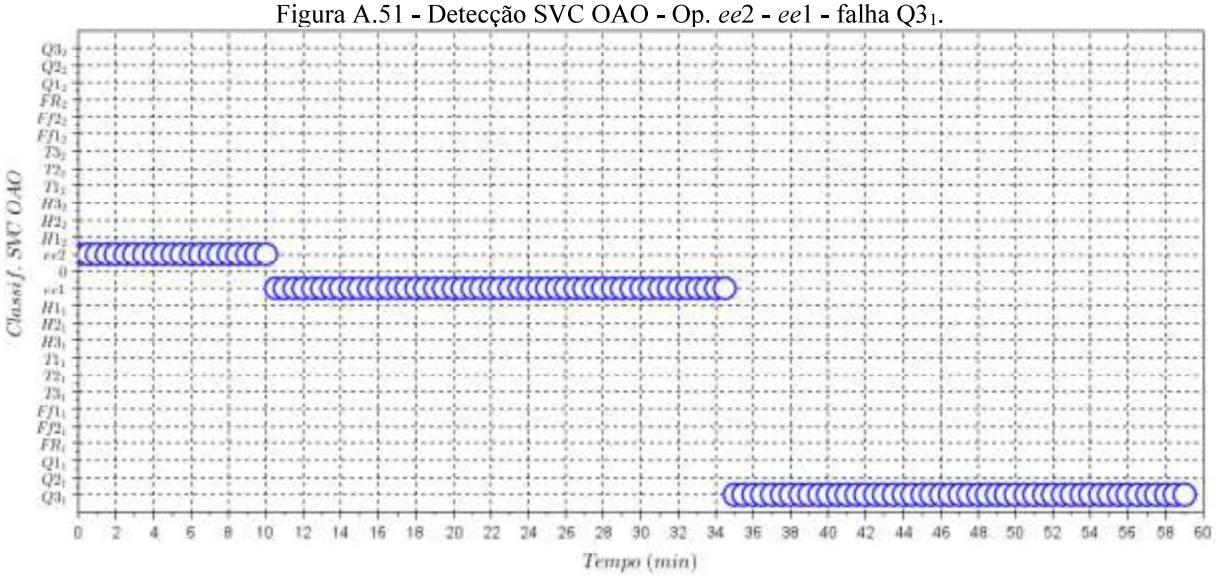
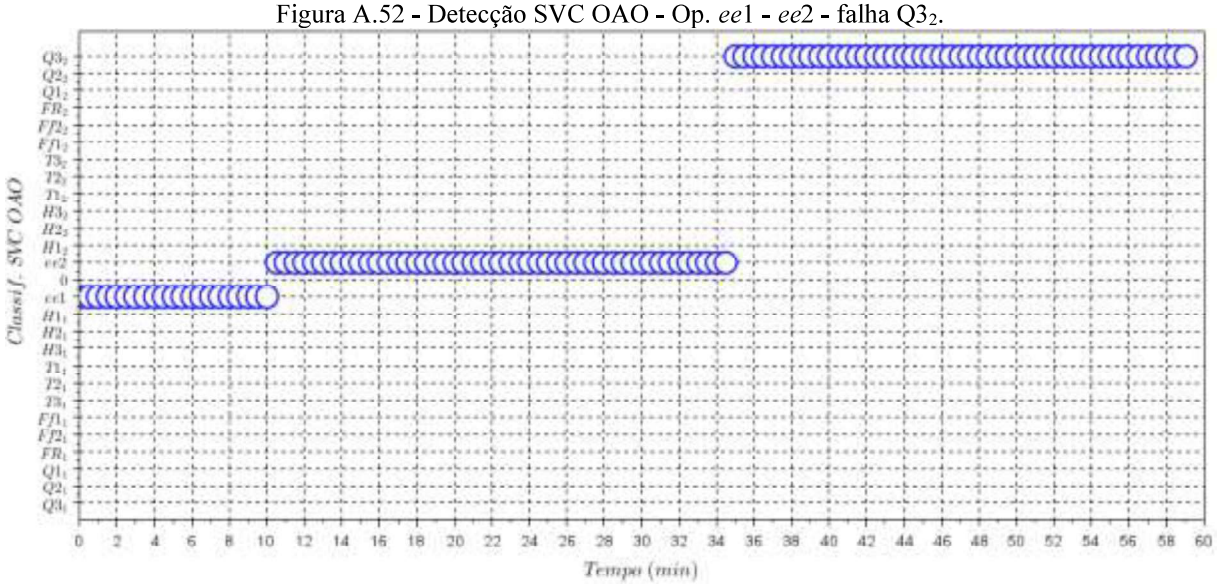


Figura A.52 - Detecção SVC OAO - Op. *ee*1 - *ee*2 - falha $Q3_2$.



O método de separação de dados através de hiperplanos apresenta grandes possibilidades na aplicação de detecção e diagnóstico de falhas. Quanto menores as amplitudes das falhas treinadas, melhor a capacidade do método SVM classificar diferentes falhas.

A.3. Resultados ANN FF

As mesmas falhas estabelecidas para o treinamento das metodologias de detecção SVC foram utilizadas para o treinamento da metodologia baseada em Redes Neuronais Artificiais *Feedforward*. As falhas foram apresentadas no Capítulo 3. A metodologia de detecção de falhas ANN FF utiliza o conjunto de dados e a matriz de probabilidades estabelecidos para os sinais de treinamento de forma a treinar uma rede composta de uma ou mais camadas de nós, chamadas camadas ocultas, que de acordo com as diferentes funções de ativação podem ser excitados ou não de acordo com os nós de entrada e os nós de saída.

Para todas as situações passíveis de detecção, foram treinados sinais em que era aplicada a falha (ou outro estado estacionário), nos instantes 5 minutos e 30 minutos, em um total de 60 minutos de operação para cada sinal treinado. Os dados então foram separados em dados de treinamento e de validação. Foi utilizado uma proporção de 2 para 1 entre os dados de treinamento e validação. A cada dois instantes utilizados para treinamento, o terceiro instante era separado para validação. Posteriormente, para os dados de teste, foi simulado outro conjunto de dados cujos instantes de aplicação de falha e modificação de estado estacionário eram diferentes dos dados treinados e validados.

Para a saída do modelo ANN FF, foram estabelecidas probabilidades para que cada dado

faça parte de cada uma das classes, sendo essas classes as operações faltosas (-13 à -2 e 2 a 13) e as operações normais nos dois estados estacionários (-1 e 1, estado estacionário 1 e estado estacionário 2, respectivamente). Para a saída do treinamento e validação, foram estabelecidas juntamente às variáveis de treinamento e validação o vetor de probabilidades $P = [P_{-13} \quad P_{-12} \quad \cdots \quad P_{-1} \quad P_1 \quad \cdots \quad P_{12} \quad P_{13}]^T$, para $P_i = 1$ se i = falha, e $P_j = 0$ se $j \neq$ falha. As falhas aplicadas -13 a -2 e 2 a 13 seguem as definições apresentadas na Tabela A.2.

Tabela A.2 - Probabilidades e falhas aplicadas no estudo de caso 2.

Índice P	Falha Variável_ee	Índice P	Falha Variável_ee	Índice P	Falha Variável_ee	Índice P	Falha Variável_ee
-13	Q3_1	-6	H2_1	1	ee2	8	Ff2_2
-12	FR_1	-5	Q1_1	2	H1_2	9	Q2_2
-11	T3_1	-4	Ff1_1	3	T1_2	10	H3_2
-10	H3_1	-3	T1_1	4	Ff1_2	11	T3_2
-9	Q2_1	-2	H1_1	5	Q1_2	12	FR_2
-8	Ff2_1	-1	ee1	6	H2_2	13	Q3_2
-7	T2_1	-	-	7	T2_2	-	-

Nas figuras a seguir, apresentam-se os resultados obtidos para o sistema de detecção utilizando redes neuronais artificiais *feedforward*, para todas as situações apresentadas na Tabela A.2.

Figura A.53 - Detecção ANN FF - Op. *ee*2 - *ee*1 - falha $Q3_1$.

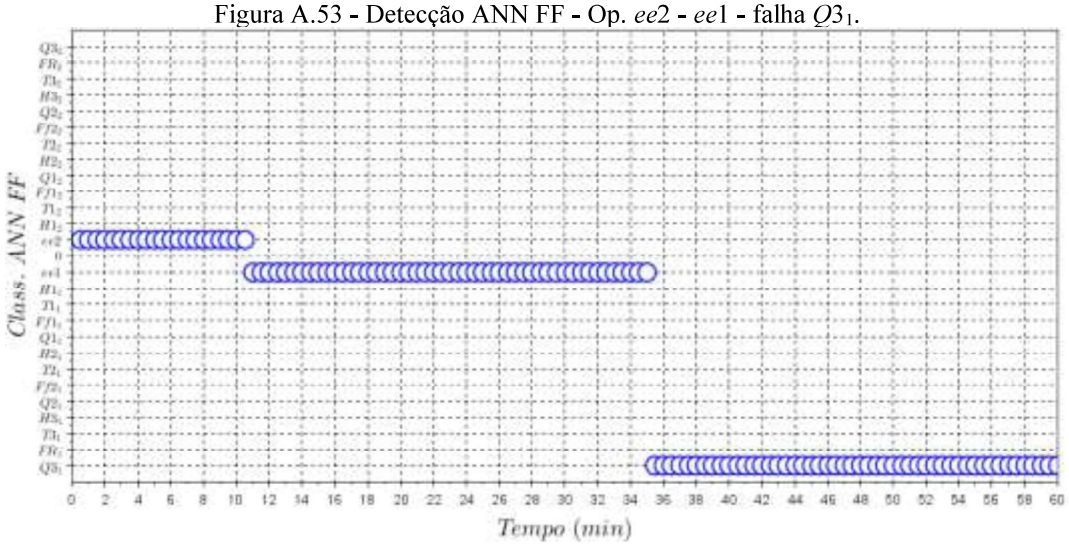


Figura A.54 - Detecção ANN FF - Op. *ee*2 - *ee*1 - falha FR_1.

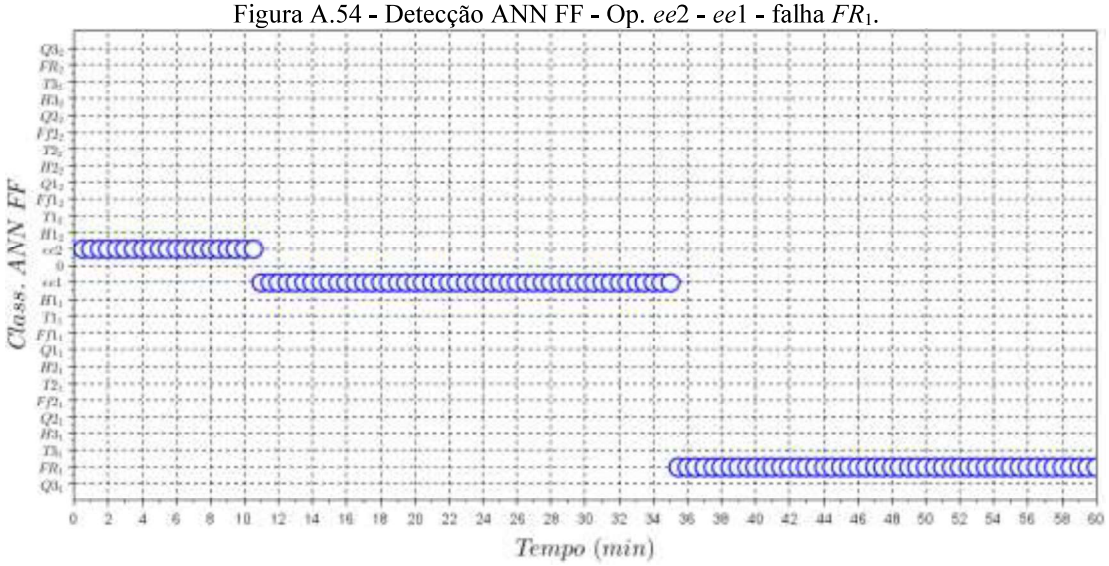


Figura A.55 - Detecção ANN FF - Op. *ee*2 - *ee*1 - falha $T3_1$.

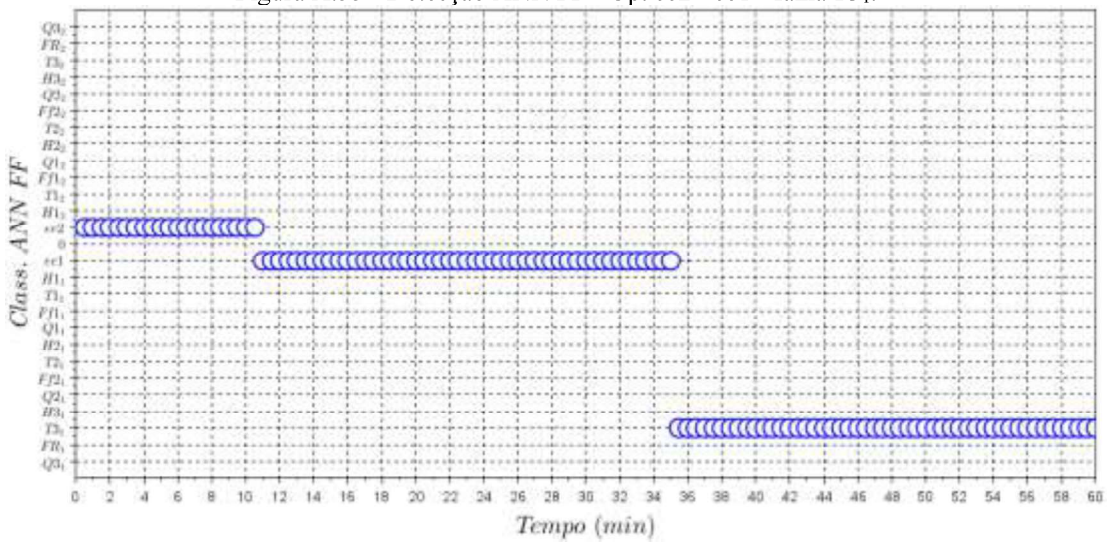


Figura A.56 - Detecção ANN FF - Op. *ee*2 - *ee*1 - falha $H3_1$.

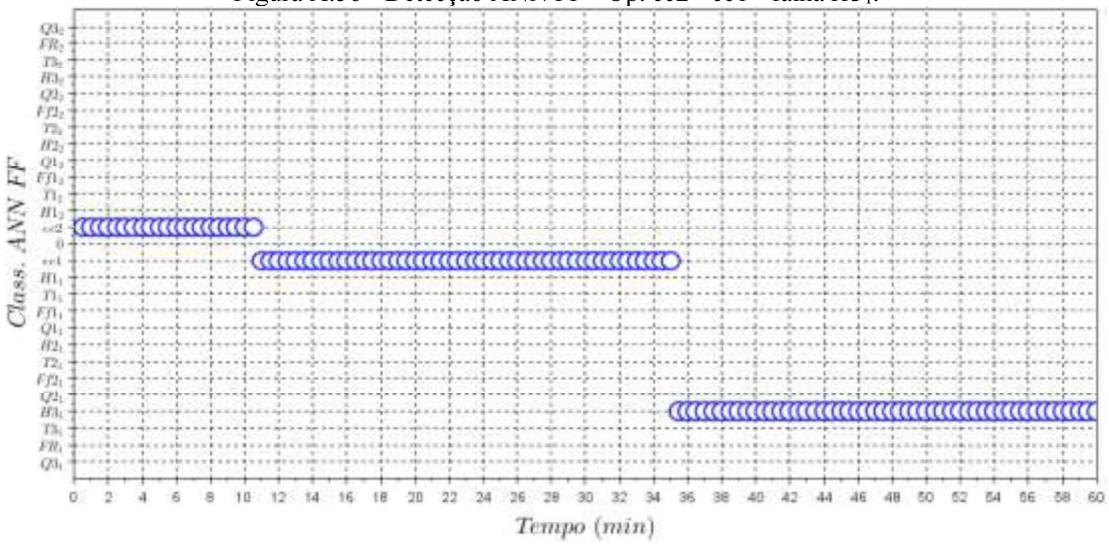


Figura A.57 - Detecção ANN FF - Op. *ee*2 - *ee*1 - falha $Q2_1$.

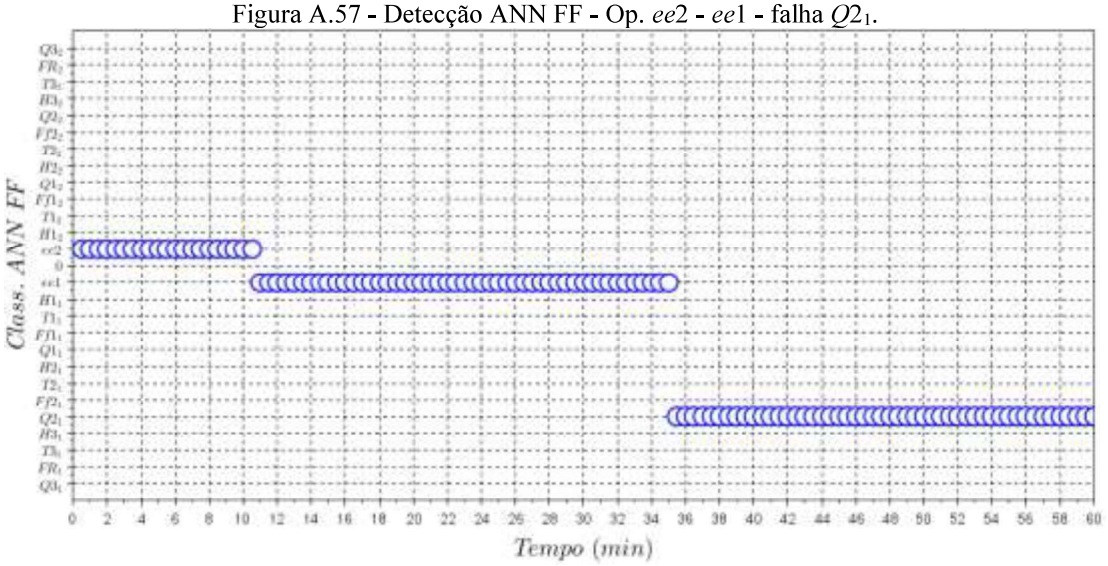


Figura A.58 - Detecção ANN FF - Op. *ee*2 - *ee*1 - falha $Ff2_1$.

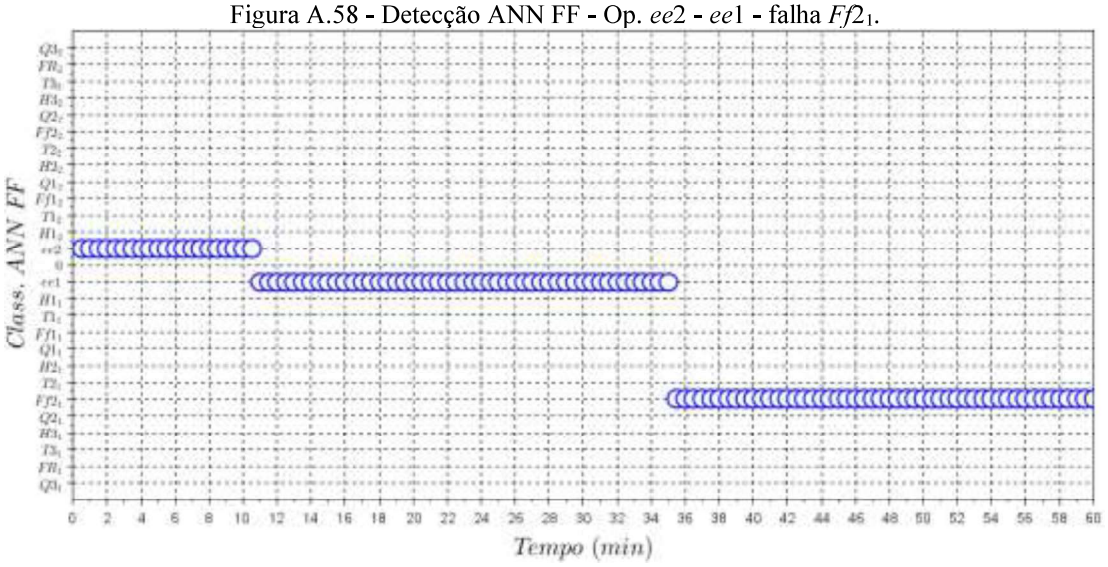


Figura A.59 - Detecção ANN FF - Op. *ee*2 - *ee*1 - falha $T2_1$.

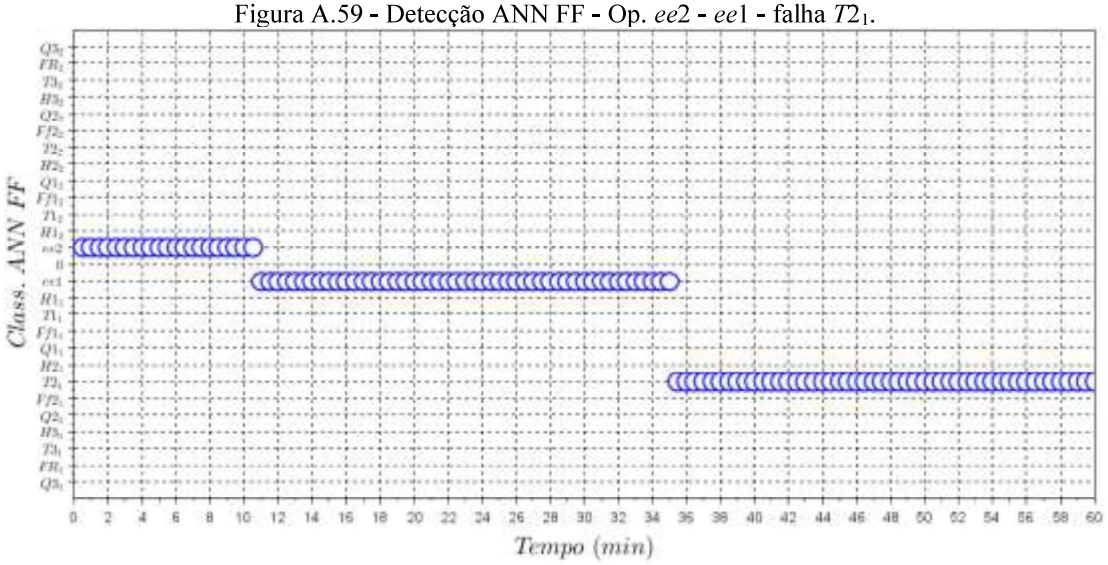


Figura A.60 - Detecção ANN FF - Op. *ee*2 - *ee*1 - falha $H2_1$.

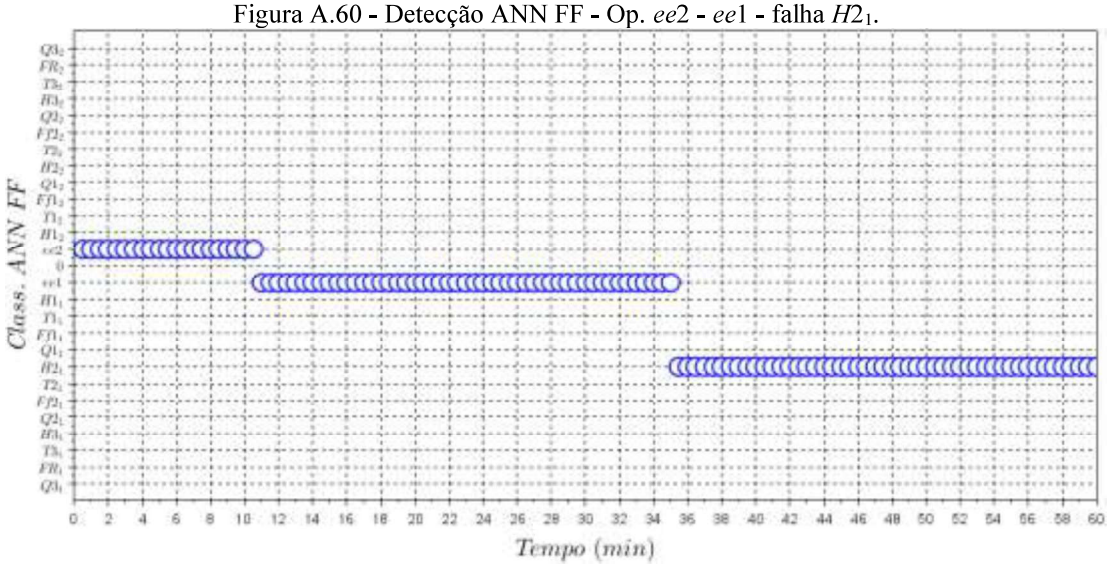


Figura A.61 - Detecção ANN FF - Op. *ee*2 - *ee*1 - falha $Q1_1$.

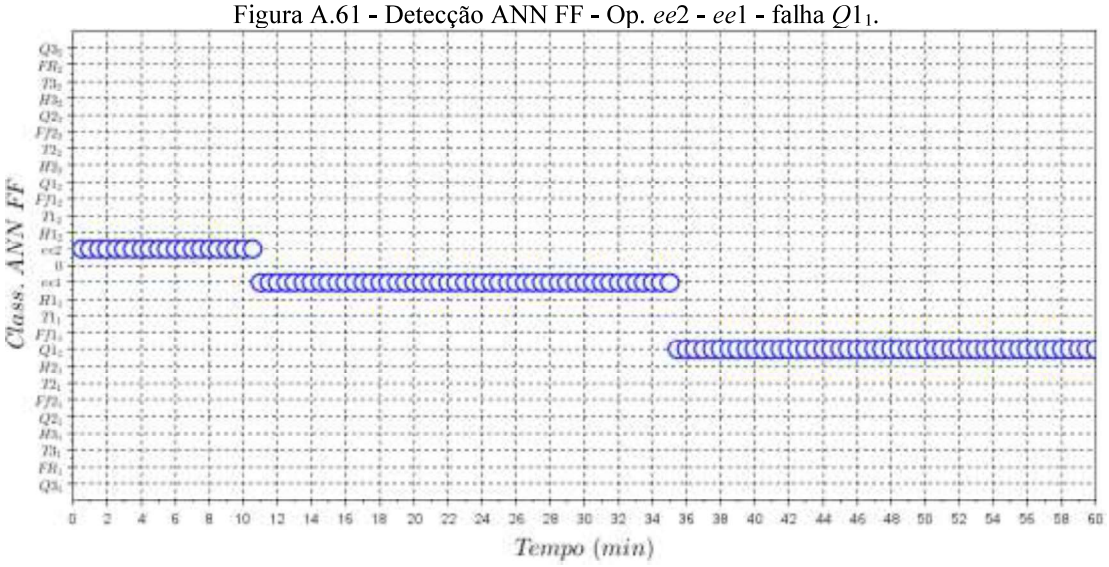


Figura A.62 - Detecção ANN FF - Op. *ee*2 - *ee*1 - falha $Ff1_1$.

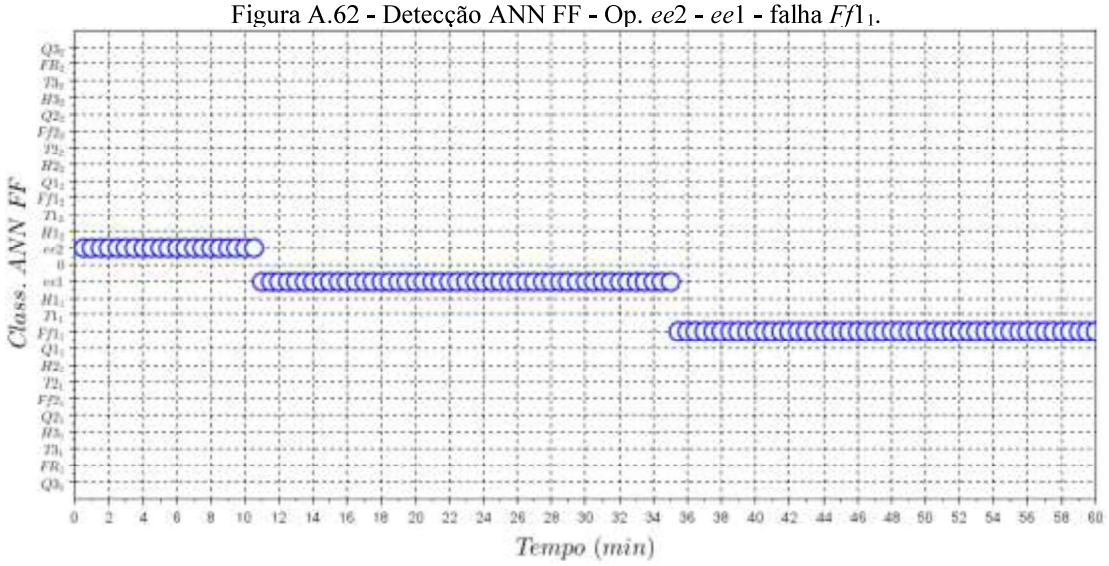


Figura A.63 - Detecção ANN FF - Op. *ee*2 - *ee*1 - falha $T1_1$.

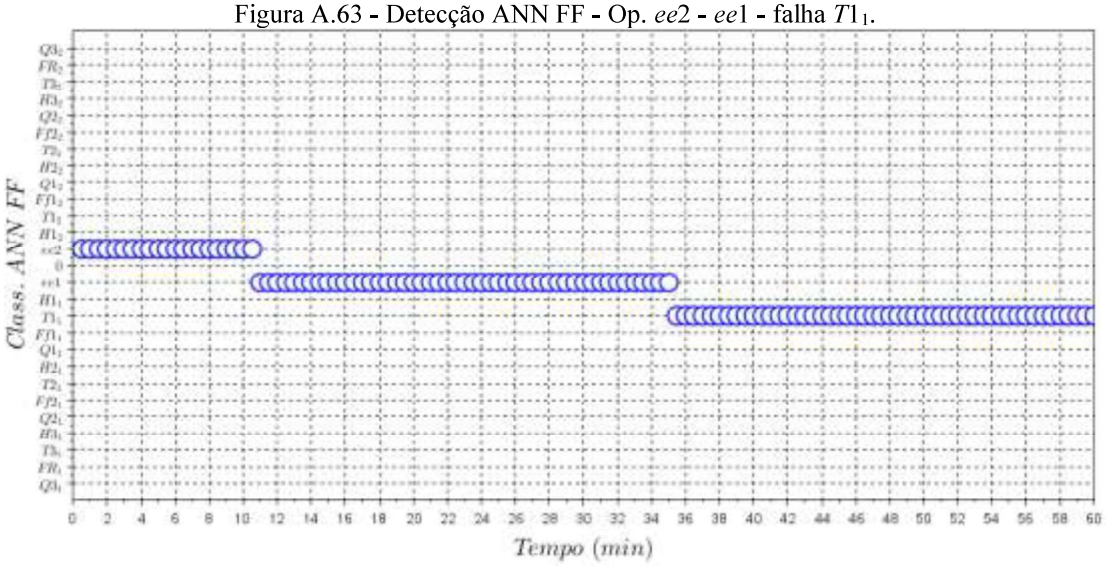


Figura A.64 - Detecção ANN FF - Op. *ee*2 - *ee*1 - falha $H1_1$.

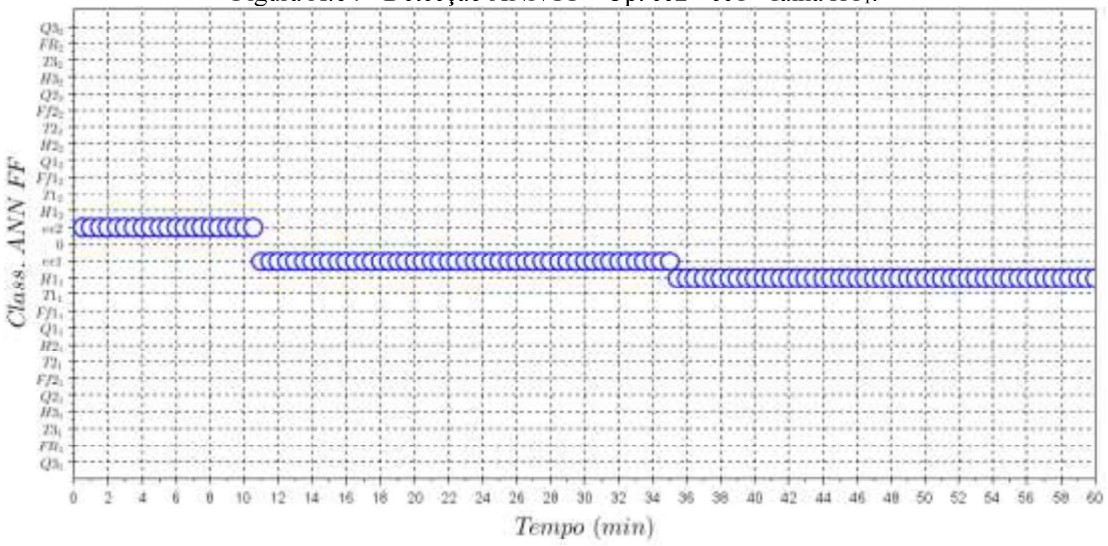


Figura A.65 - Detecção ANN FF - Op. *ee*2 - *ee*1 - *ee*2.

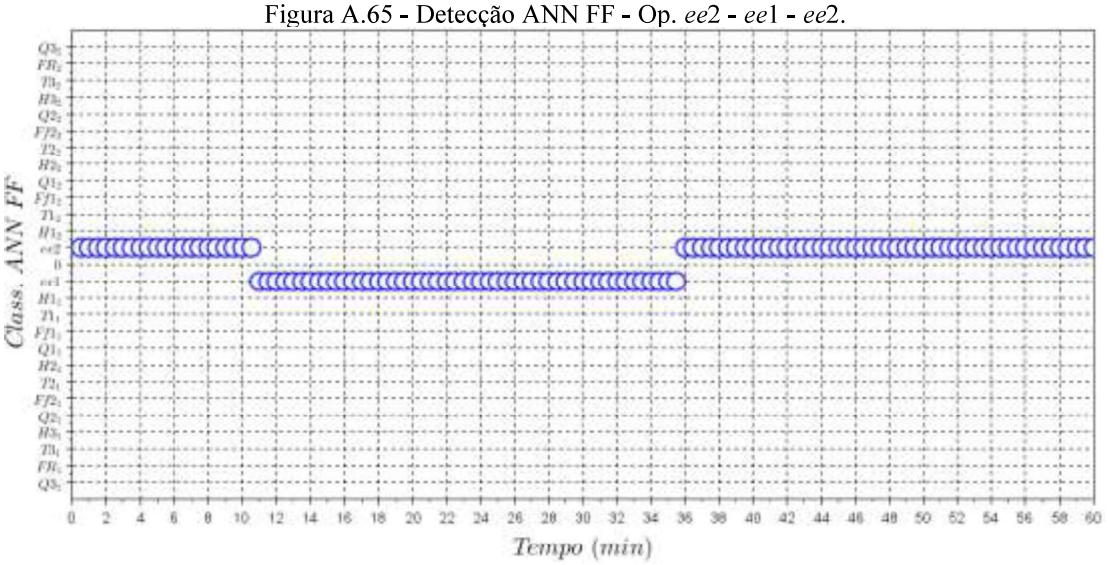


Figura A.66 - Detecção ANN FF - Op. *ee*1 - *ee*2 - *ee*1.

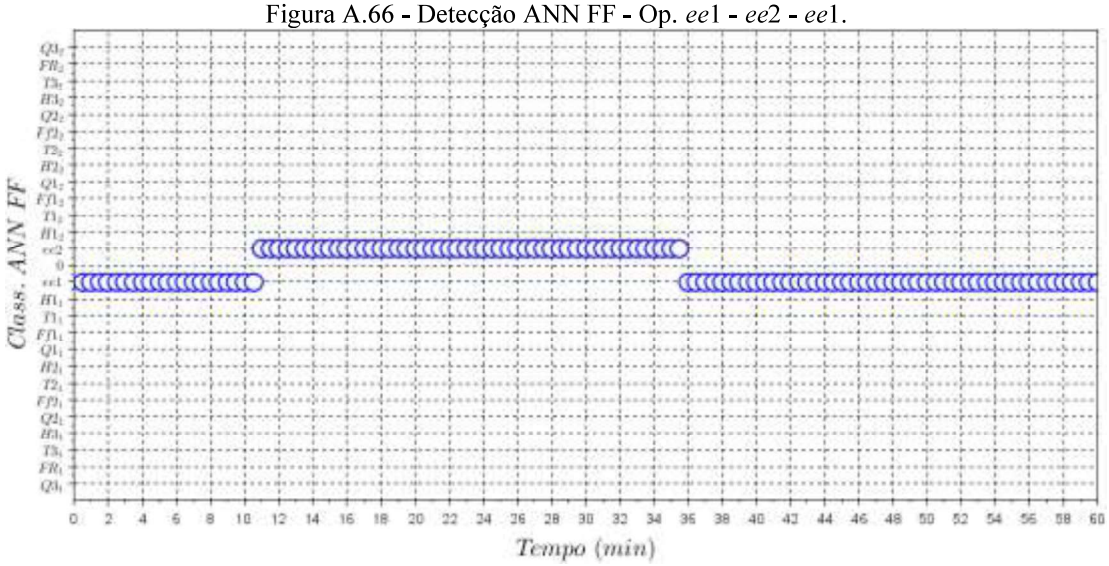


Figura A.67 - Detecção ANN FF - Op. *ee*1 - *ee*2 - falha $H1_2$.

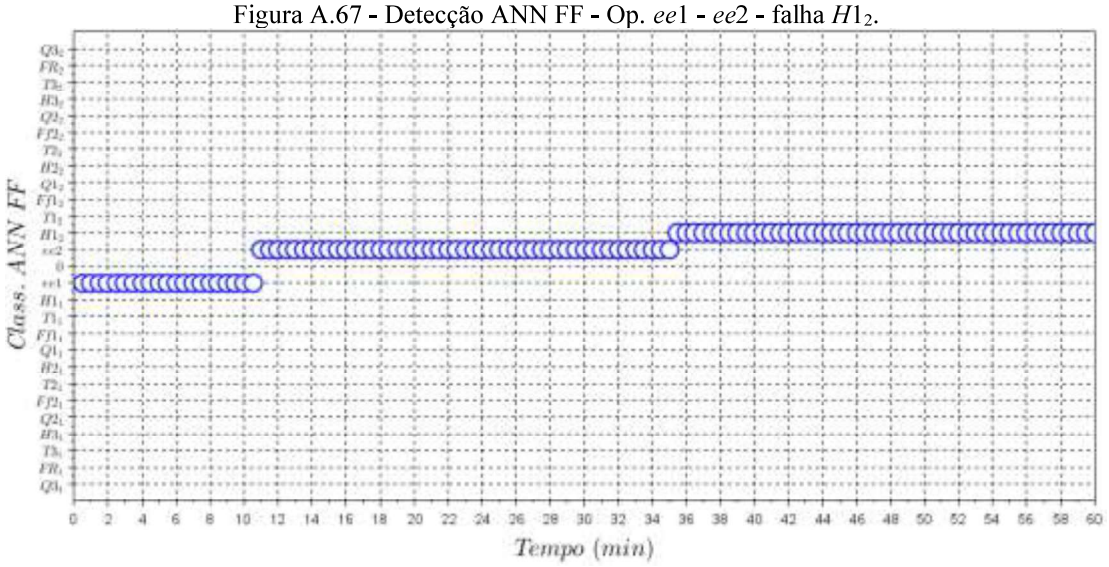


Figura A.68 - Detecção ANN FF - Op. *ee*1 - *ee*2 - falha $T1_2$.

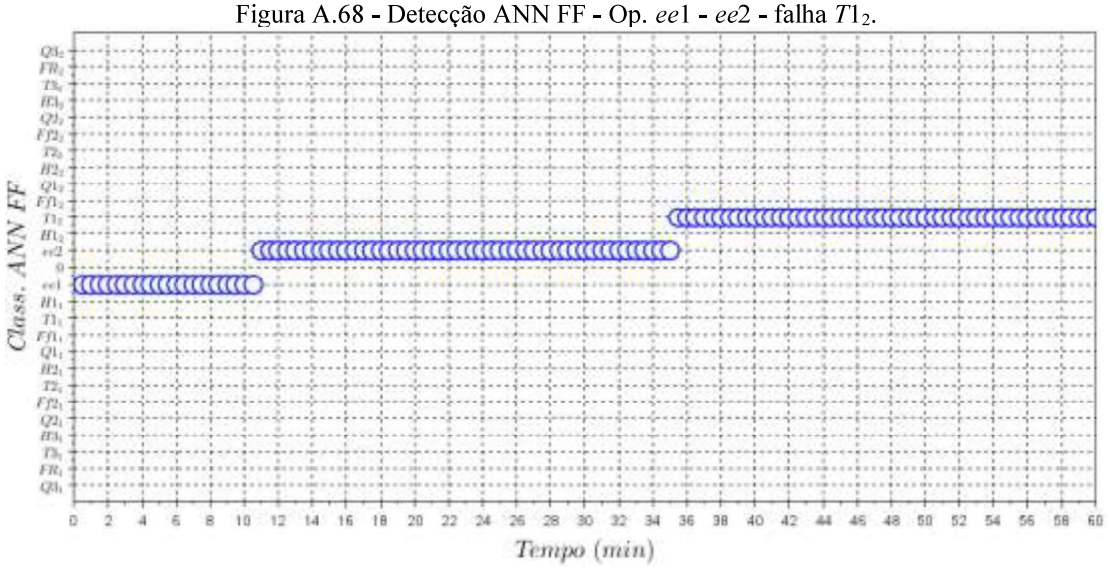


Figura A.69 - Detecção ANN FF - Op. *ee*1 - *ee*2 - falha $Ff1_2$.

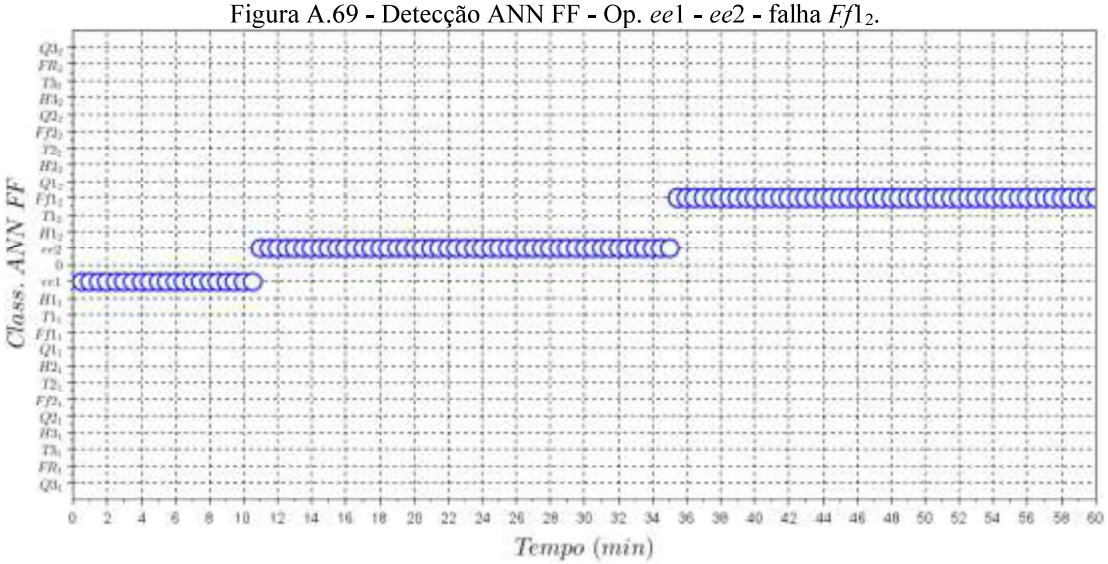


Figura A.70 - Detecção ANN FF - Op. *ee*1 - *ee*2 - falha $Q1_2$.

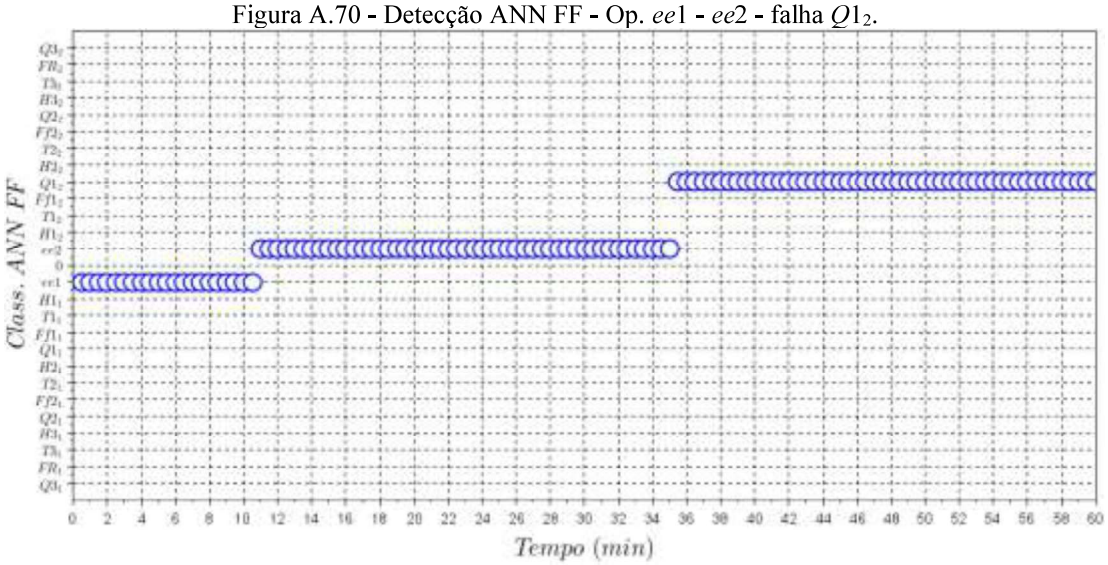


Figura A.71 - Detecção ANN FF - Op. *ee*1 - *ee*2 - falha $H2_2$.

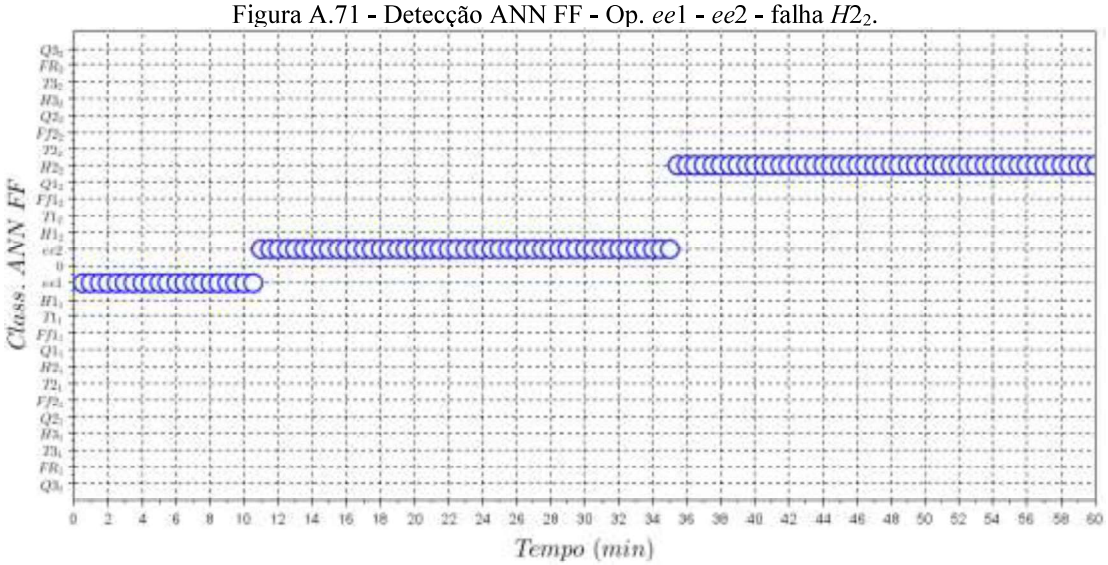


Figura A.72 - Detecção ANN FF - Op. *ee*1 - *ee*2 - falha $T2_2$.

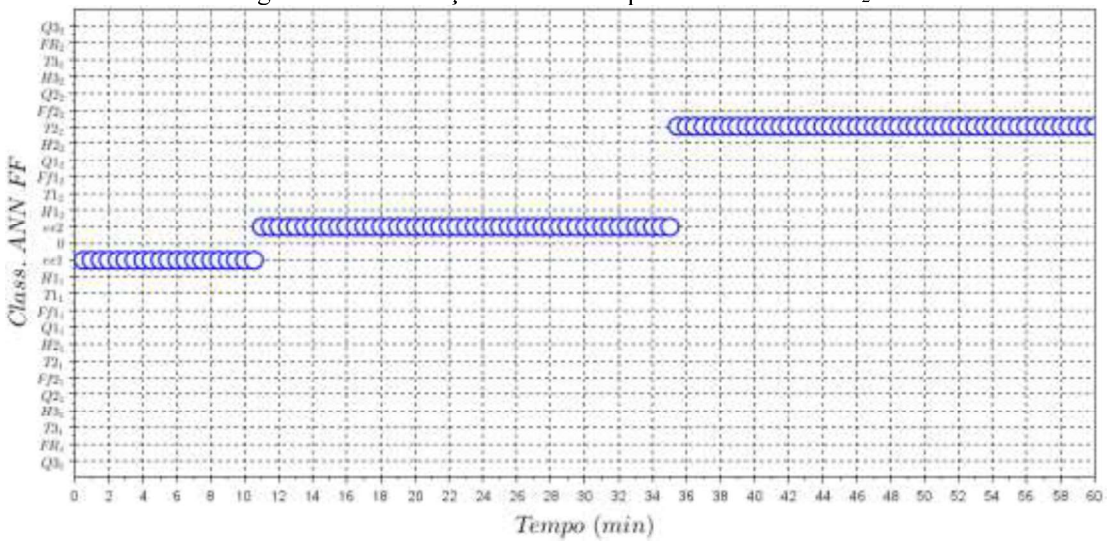


Figura A.73 - Detecção ANN FF - Op. *ee*1 - *ee*2 - falha $Ff2_2$.

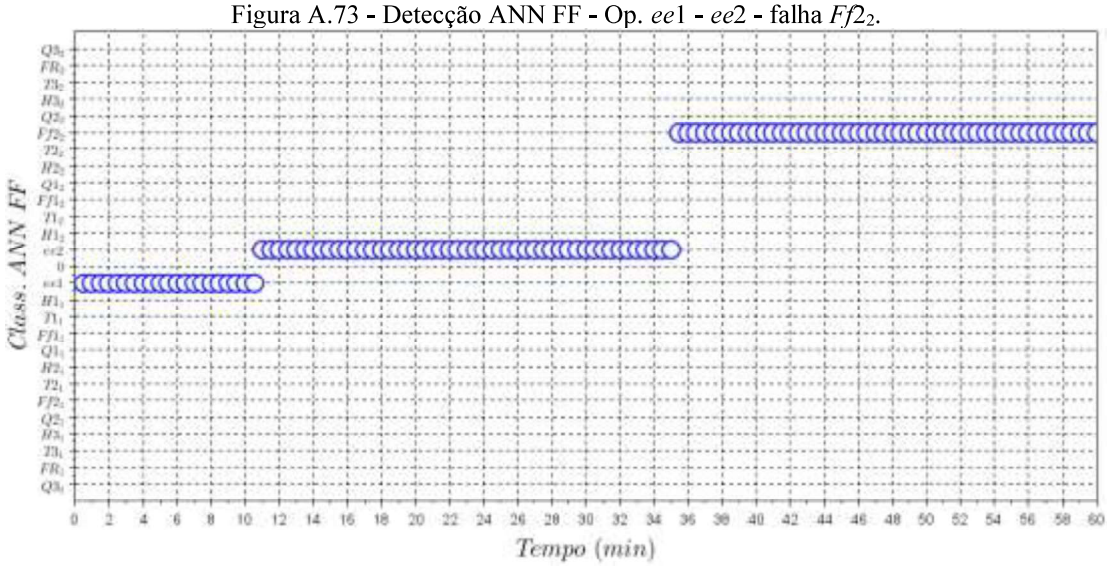


Figura A.74 - Detecção ANN FF - Op. *ee*1 - *ee*2 - falha $Q2_2$.

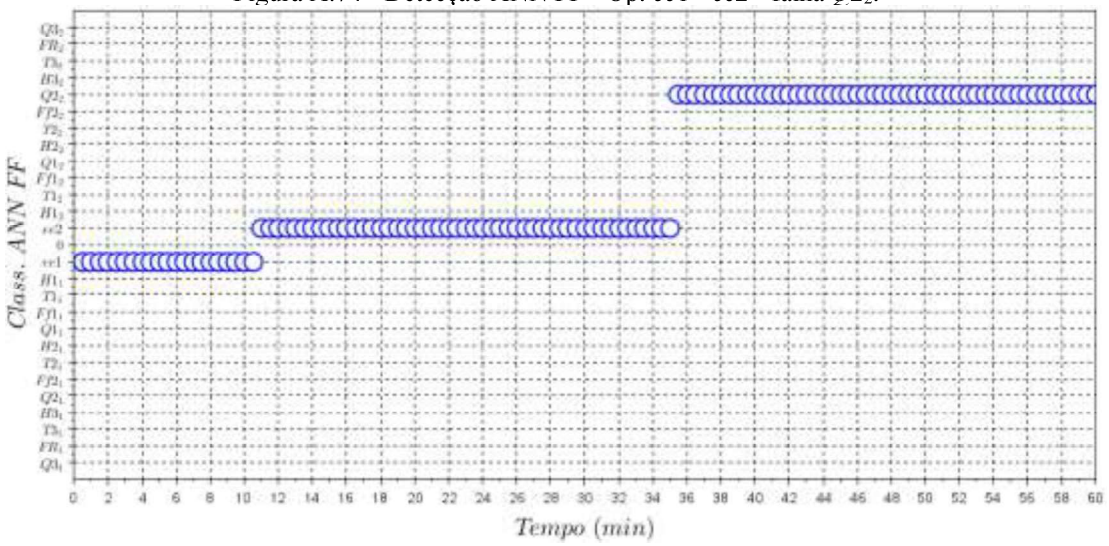


Figura A.75 - Detecção ANN FF - Op. *ee*1 - *ee*2 - falha $H3_2$.

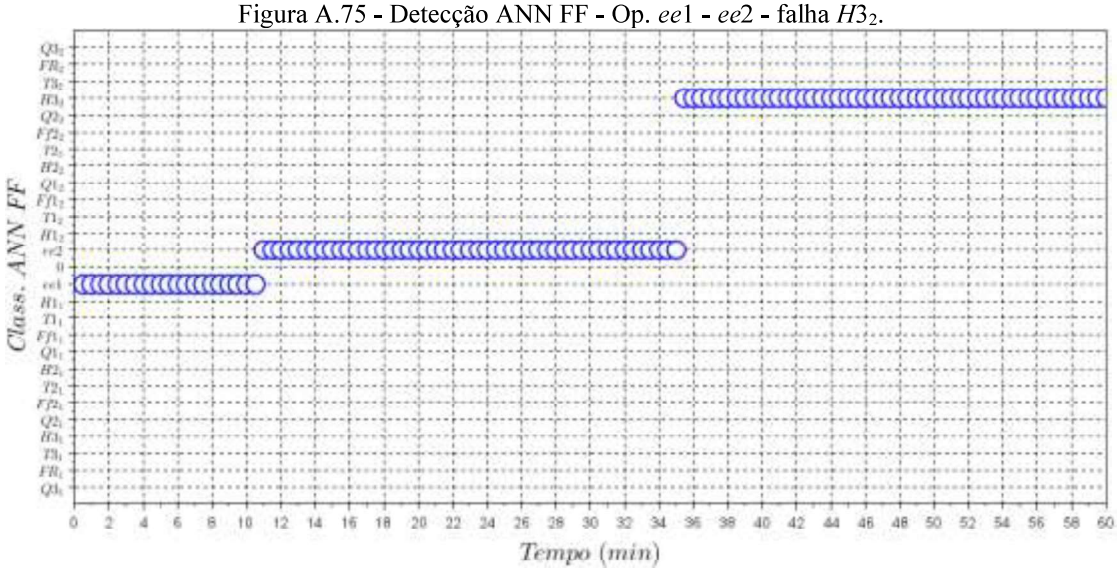


Figura A.76 - Detecção ANN FF - Op. *ee*1 - *ee*2 - falha $T3_2$.

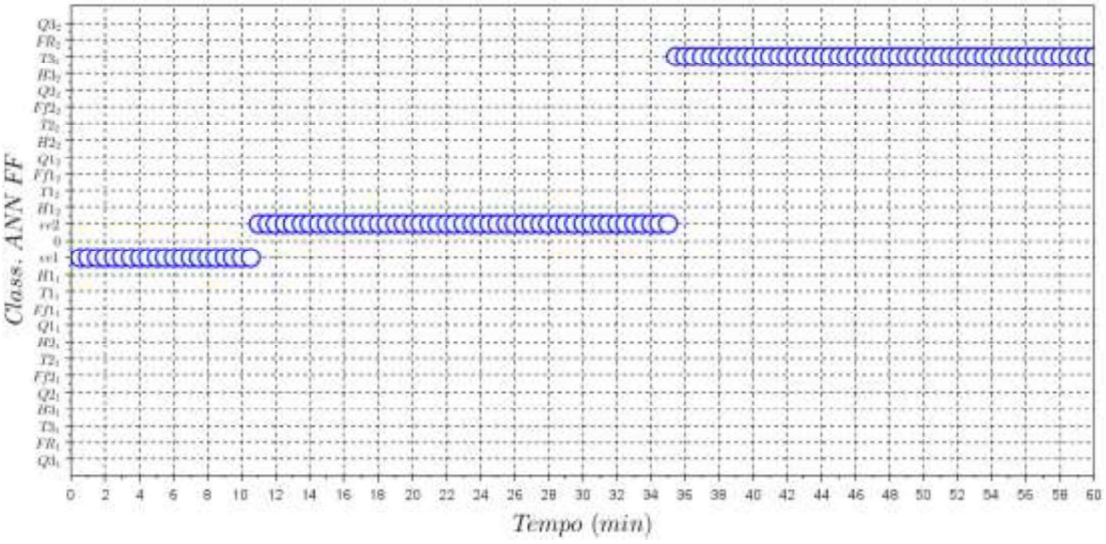


Figura A.77 - Detecção ANN FF - Op. *ee*1 - *ee*2 - falha FR_2.

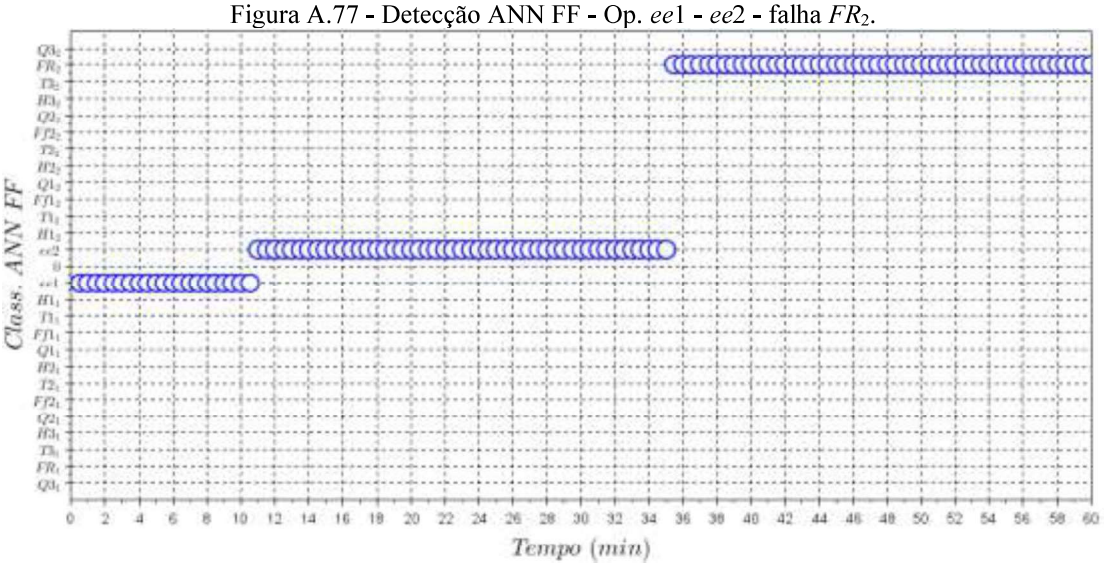
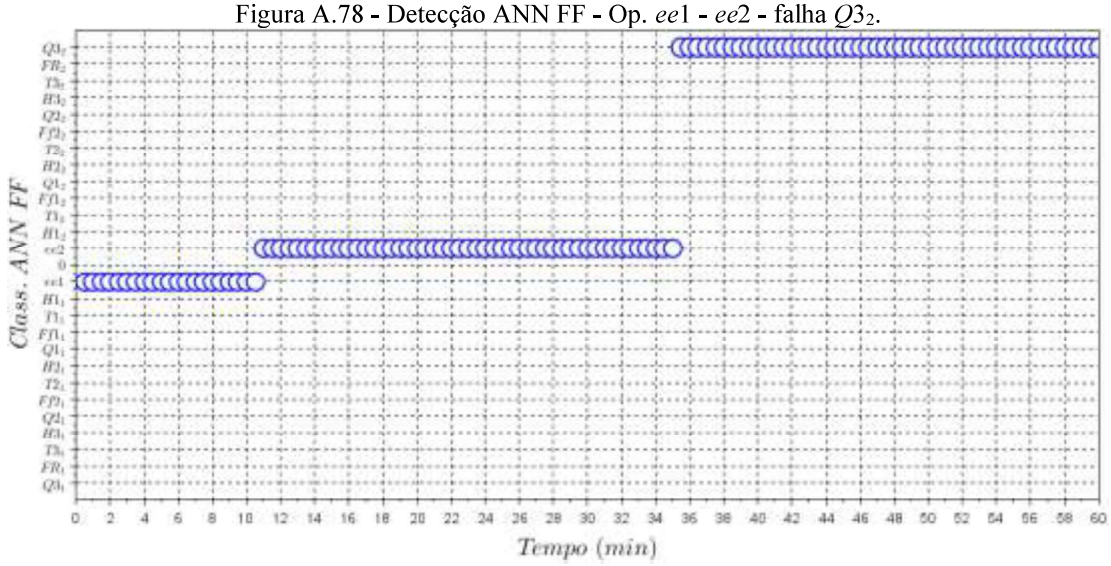


Figura A.78 - Detecção ANN FF - Op. *ee*1 - *ee*2 - falha $Q3_2$.



A.4. Resultados ANN SOM

As mesmas falhas estabelecidas para o treinamento das metodologias de detecção SVC foram utilizadas para o treinamento da metodologia baseada em Redes Neuronais Artificiais *Self-organizing Map* (SOM), ou Mapa Auto-organizável. As falhas foram apresentadas no Capítulo 3. A metodologia de detecção de falhas ANN SOM utiliza somente o conjunto de dados de treinamento, já que o método SOM é baseado em aprendizado não-supervisionado. Os métodos com aprendizado não-supervisionado usam somente os dados de entrada do sistema de detecção de falhas (variáveis manipuladas e/ou controladas) e informações a respeito da quantidade de classes que se deseja classificar esses dados. Para o modelo utilizado neste trabalho, foi estabelecida uma estrutura com 26 classes.

Para todas as situações passíveis de detecção, foram treinados sinais em que era aplicada a falha (ou outro estado estacionário), no instante 30 minutos, em um total de 60 minutos de operação para cada sinal treinado. Foi utilizado uma proporção de 2 para 1 entre os dados de treinamento e validação. A cada dois instantes utilizados para treinamento, o terceiro instante era separado para validação. Para os dados de teste, foi simulado outro conjunto de dados cujos instantes de aplicação de falha e modificação de estado estacionário eram diferentes dos dados treinados e validados.

Figura A.79 - Detecção ANN SOM - Op. *ee*2 - *ee*1 - *ee*2.

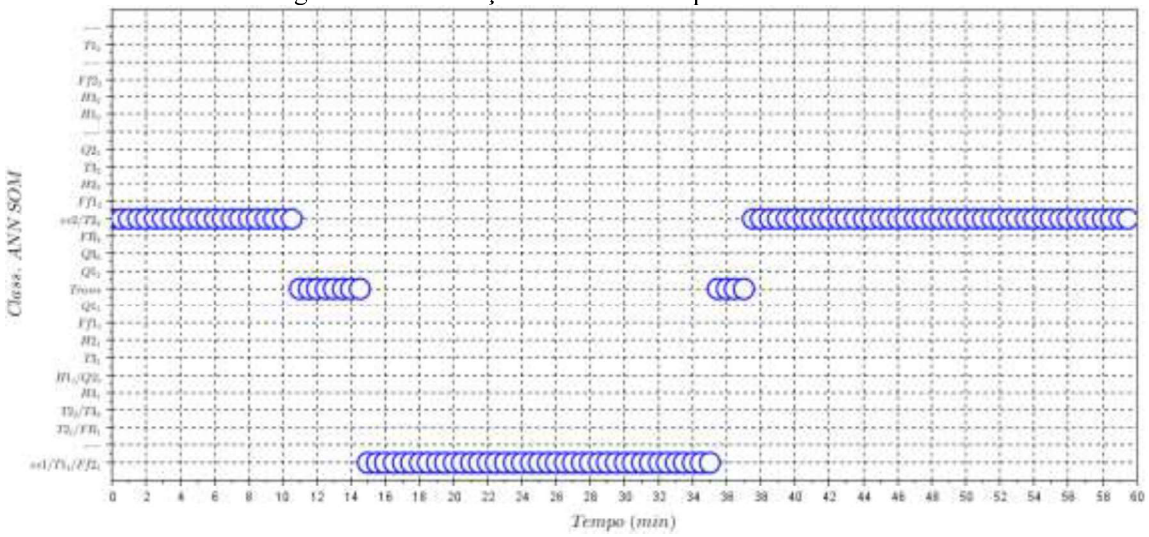


Figura A.80 - Detecção ANN SOM - Op. *ee*1 - *ee*2 - *ee*1.

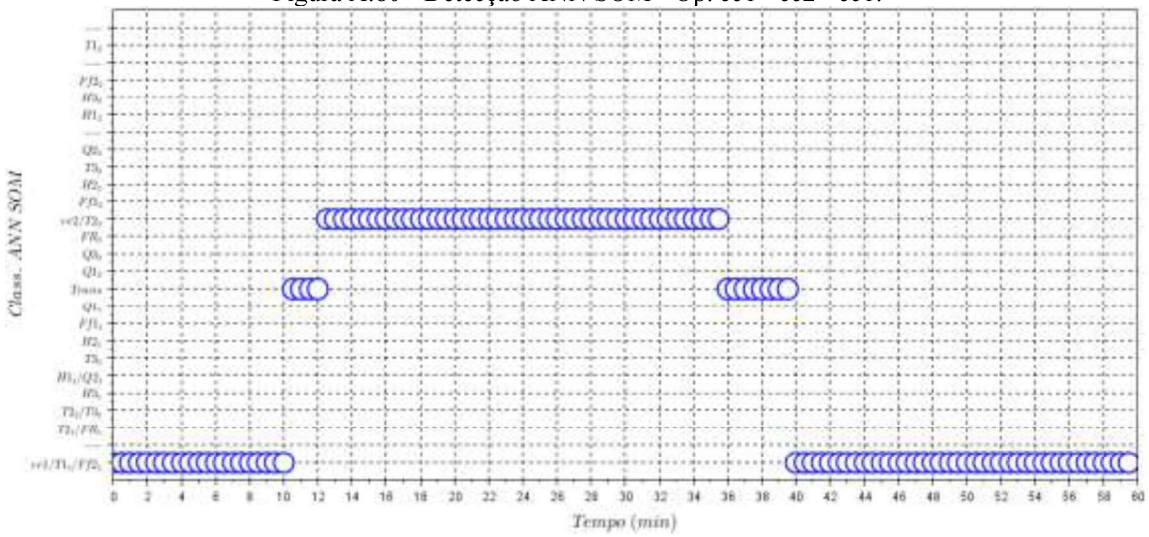


Figura A.81 - Detecção ANN SOM - Op. *ee*2 - *ee*1 - falha $H1_1$.

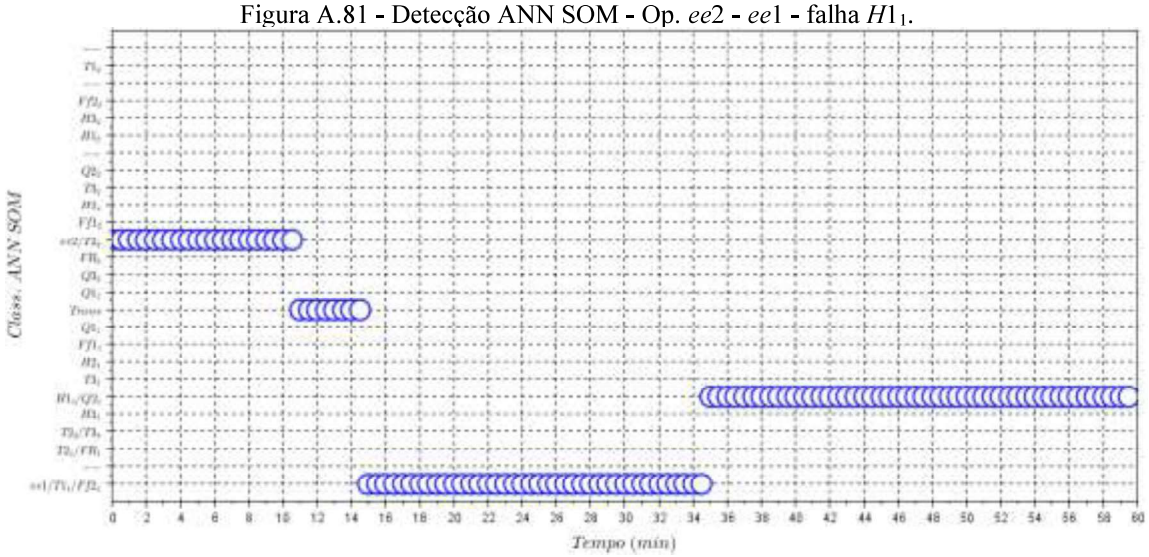


Figura A.82 - Detecção ANN SOM - Op. *ee*1 - *ee*2 - falha $H1_2$.

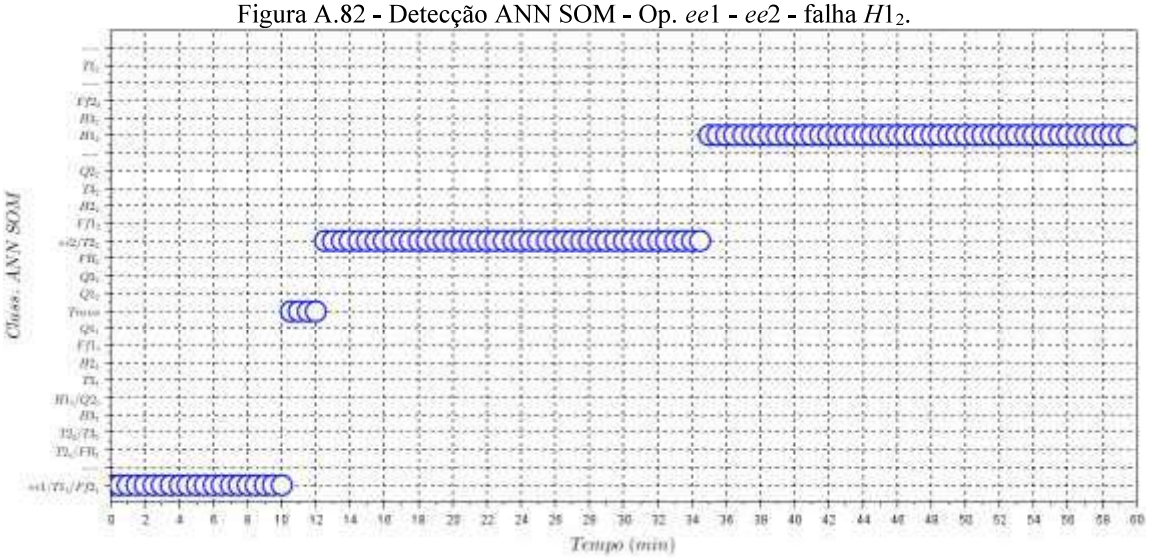


Figura A.83 - Detecção ANN SOM - Op. *ee*2 - *ee*1 - falha $H2_1$.

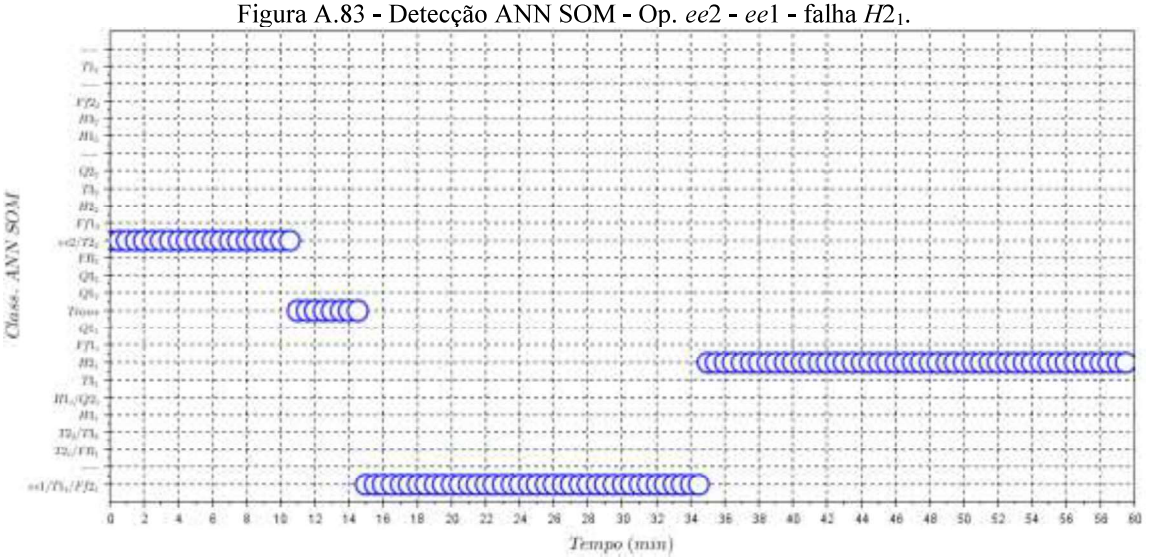


Figura A.84 - Detecção ANN SOM - Op. *ee*1 - *ee*2 - falha $H2_2$.

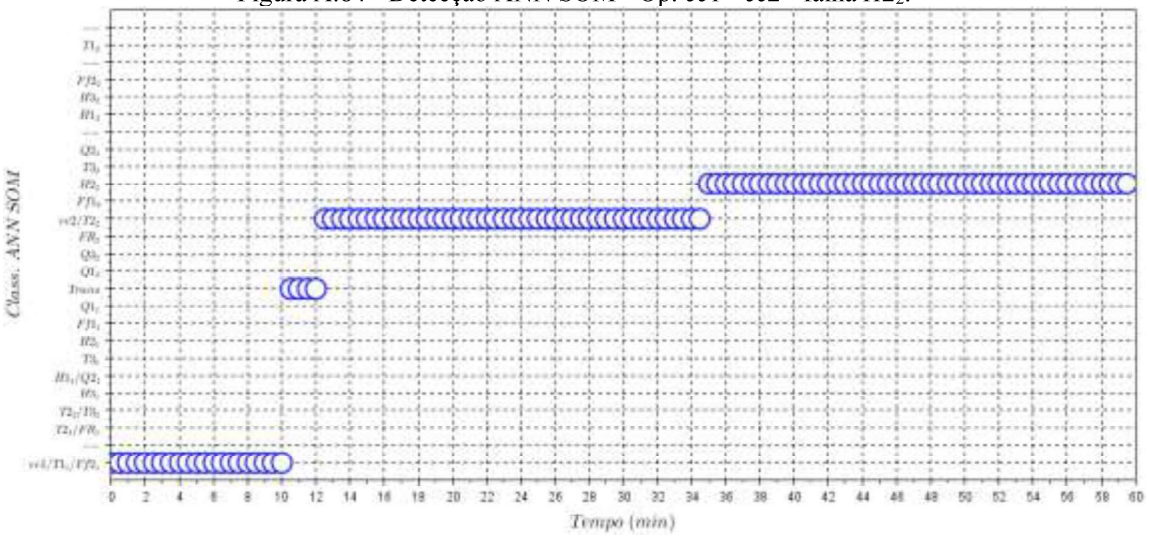


Figura A.85 - Detecção ANN SOM - Op. *ee*2 - *ee*1 - falha $H3_1$.

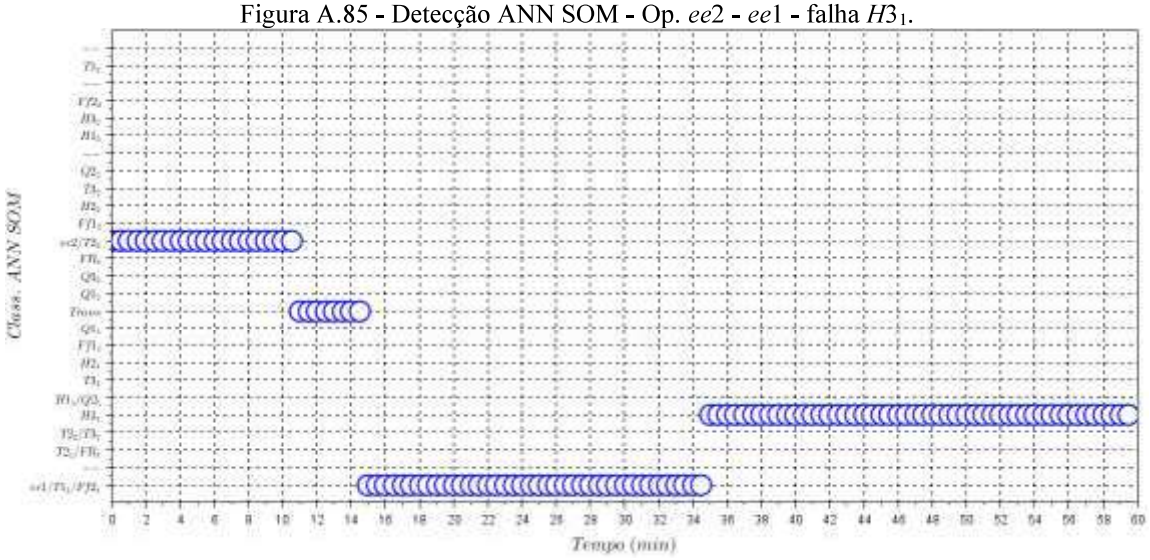


Figura A.86 - Detecção ANN SOM - Op. *ee*1 - *ee*2 - falha $H3_2$.

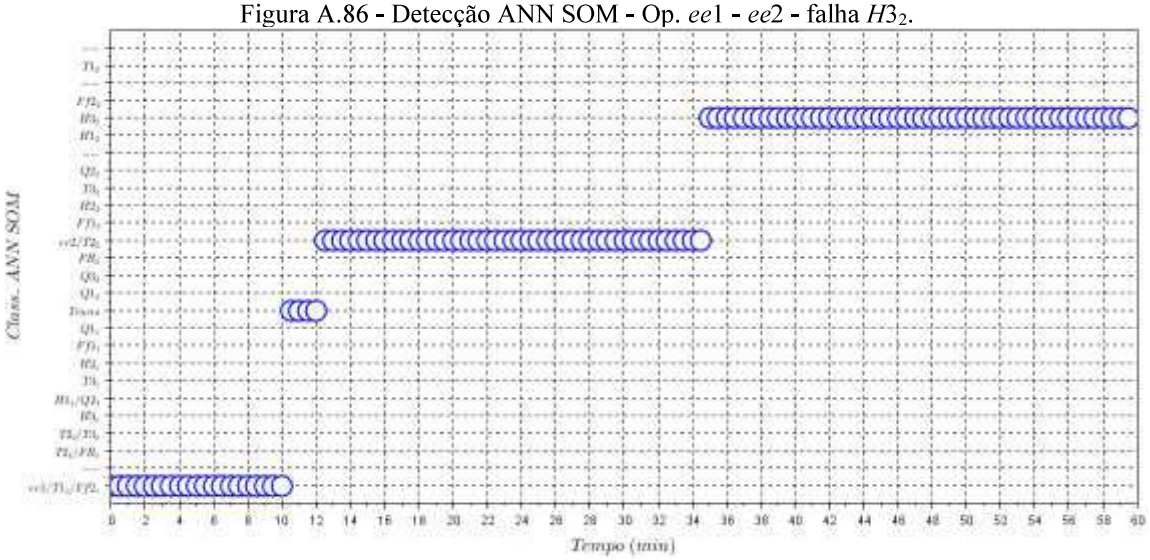


Figura A.87 - Detecção ANN SOM - Op. *ee*2 - *ee*1 - falha $T1_1$.

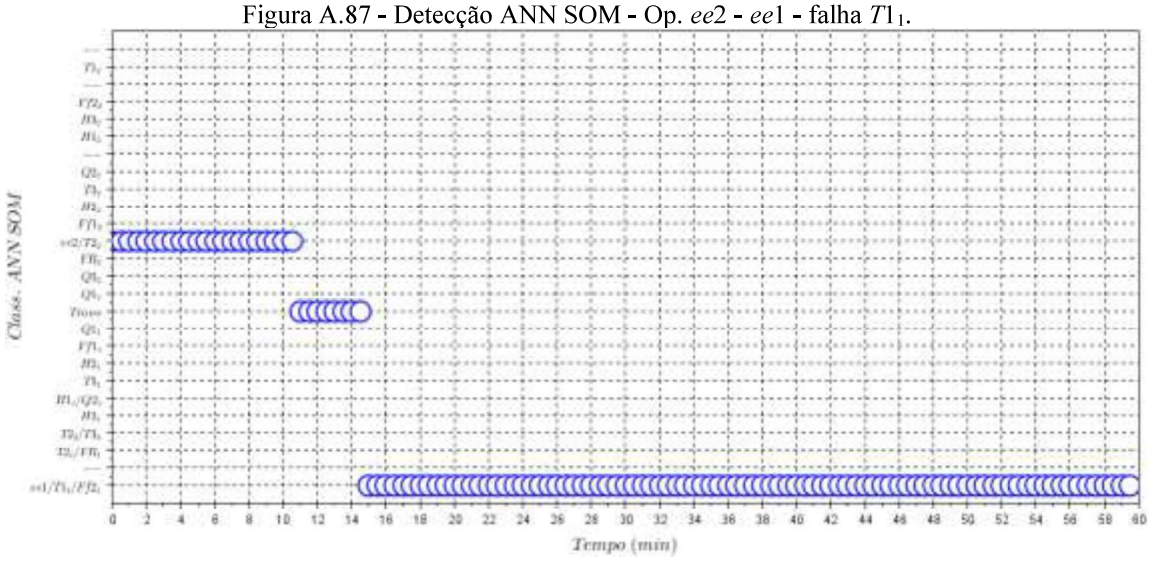


Figura A.88 - Detecção ANN SOM - Op. *ee*1 - *ee*2 - falha $T1_2$.

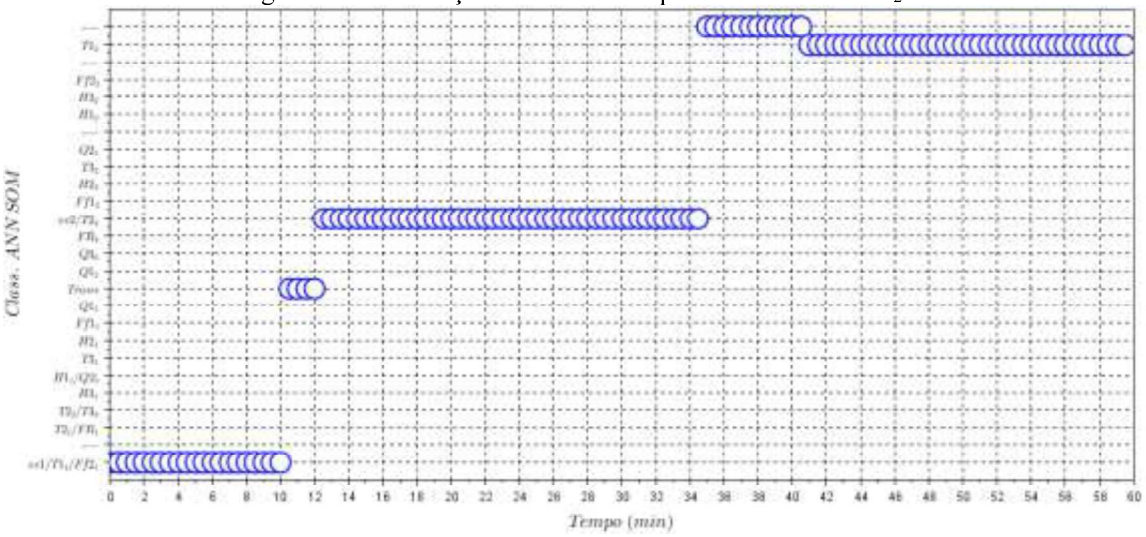


Figura A.89 - Detecção ANN SOM - Op. *ee*2 - *ee*1 - falha $T2_1$.

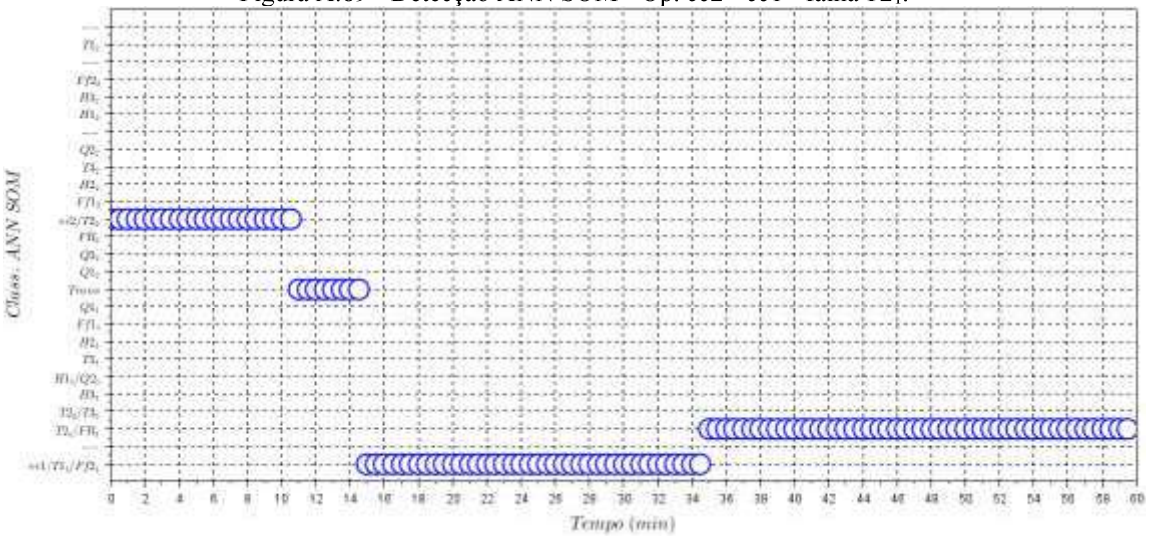


Figura A.90 - Detecção ANN SOM - Op. *ee*1 - *ee*2 - falha $T2_2$.

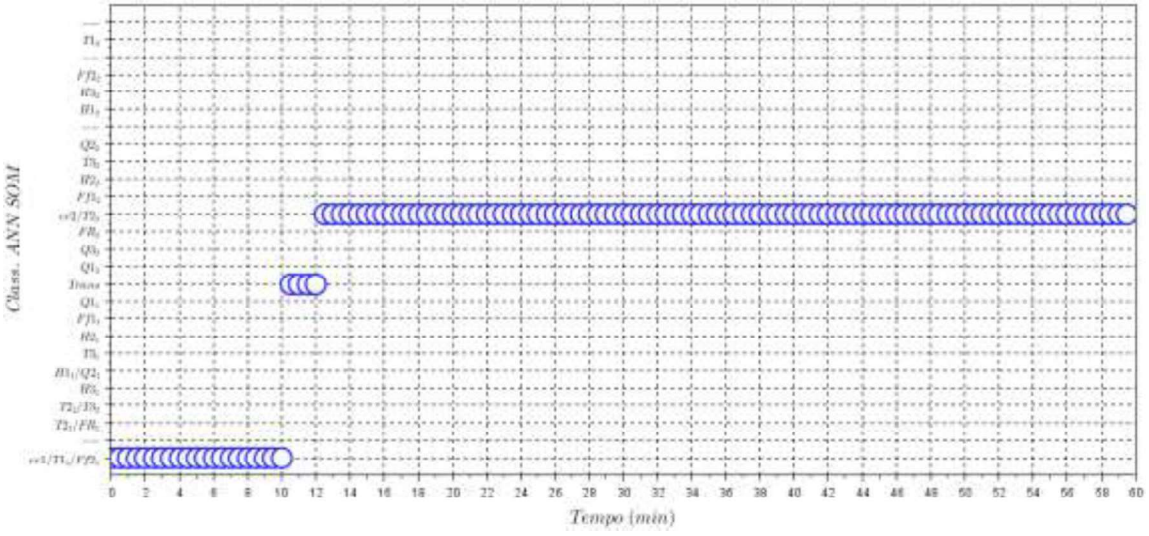


Figura A.91 - Detecção ANN SOM - Op. *ee*2 - *ee*1 - falha $T3_1$.

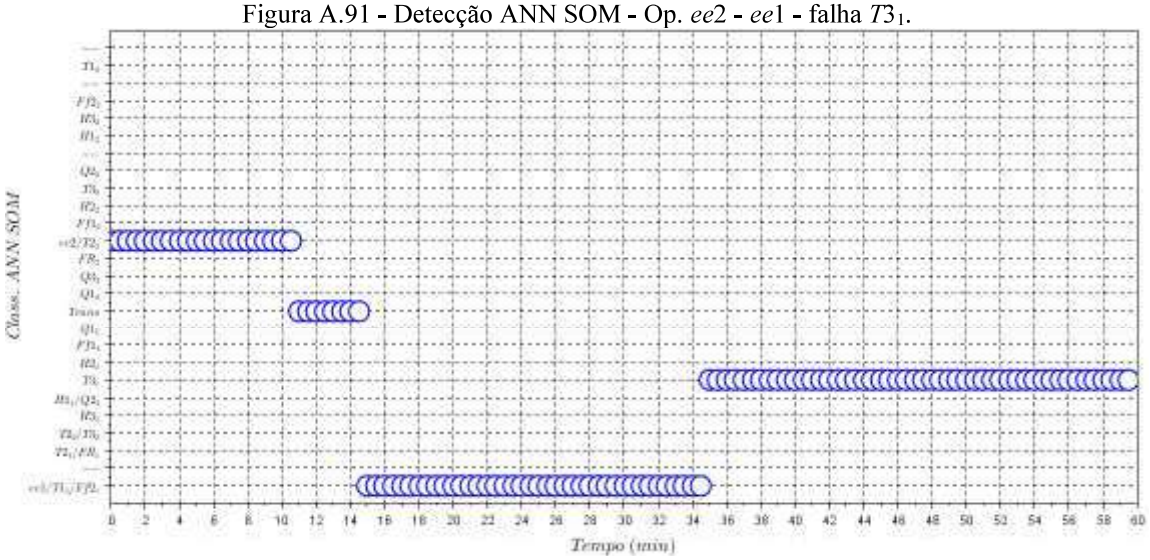


Figura A.92 - Detecção ANN SOM - Op. *ee*1 - *ee*2 - falha $T3_2$.

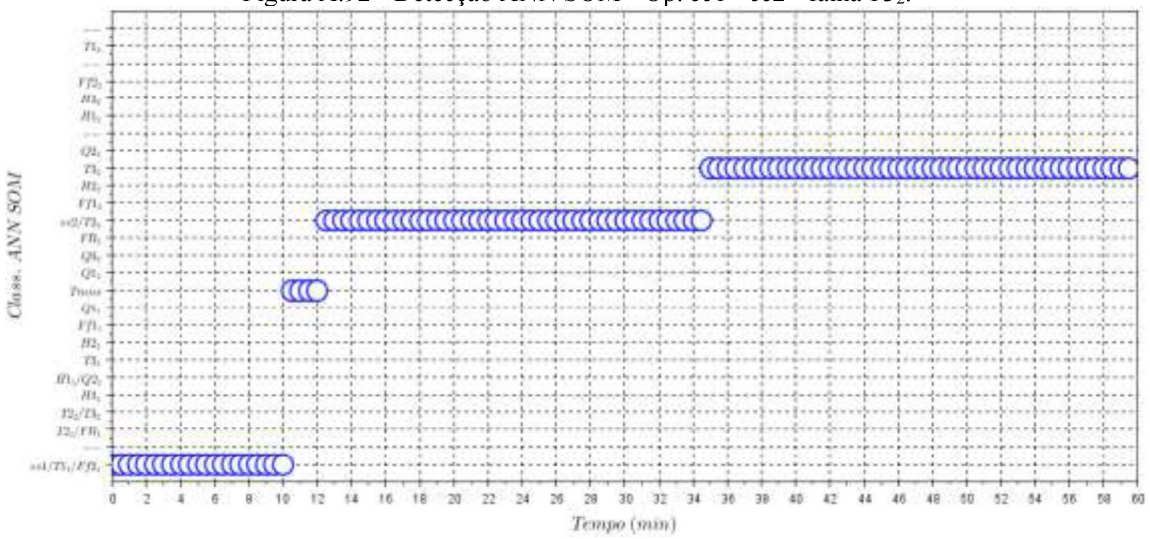


Figura A.93 - Detecção ANN SOM - Op. *ee*2 - *ee*1 - falha $Ff1_1$.

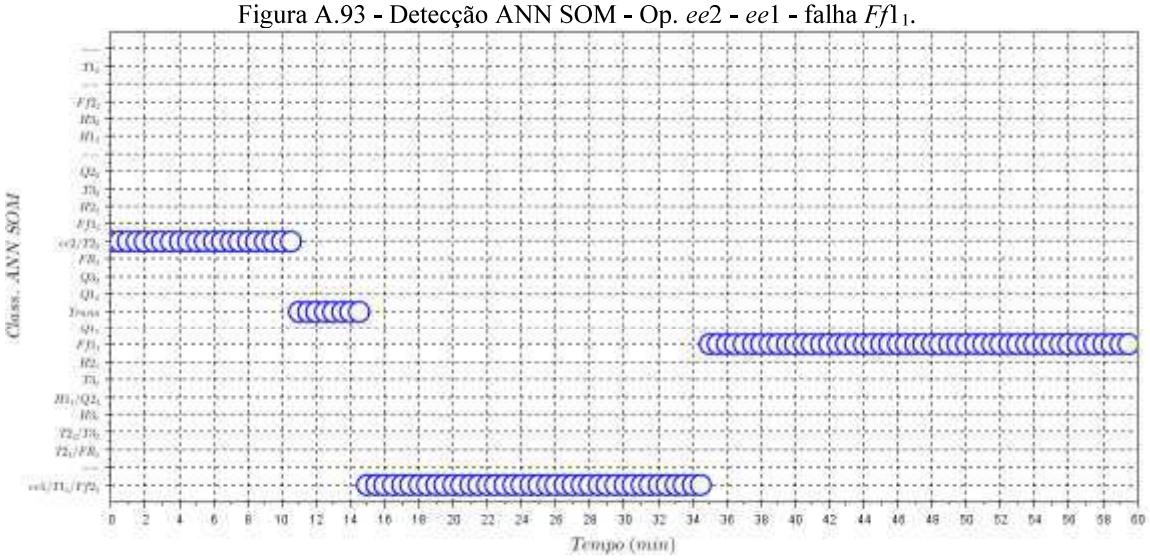


Figura A.94 - Detecção ANN SOM - Op. *ee*1 - *ee*2 - falha $Ff1_2$.

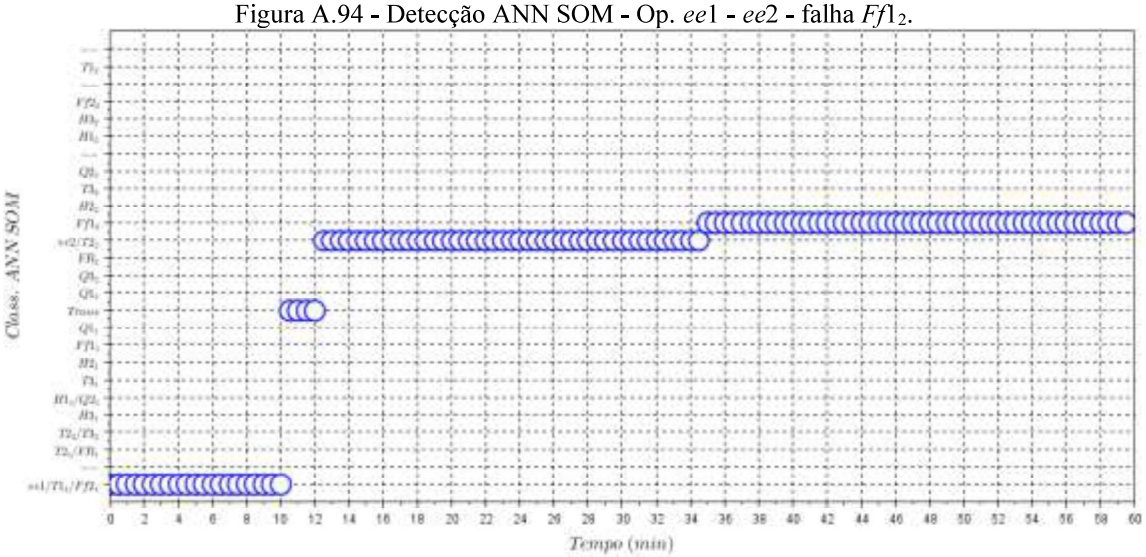


Figura A.95 - Detecção ANN SOM - Op. *ee*2 - *ee*1 - falha $Ff2_1$.

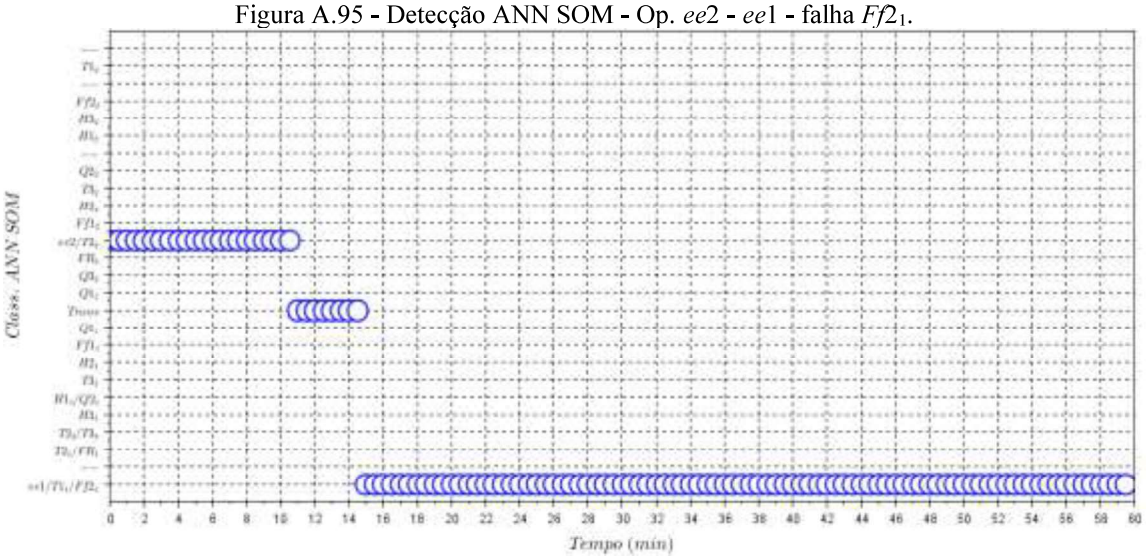


Figura A.96 - Detecção ANN SOM - Op. *ee*1 - *ee*2 - falha $Ff2_2$.

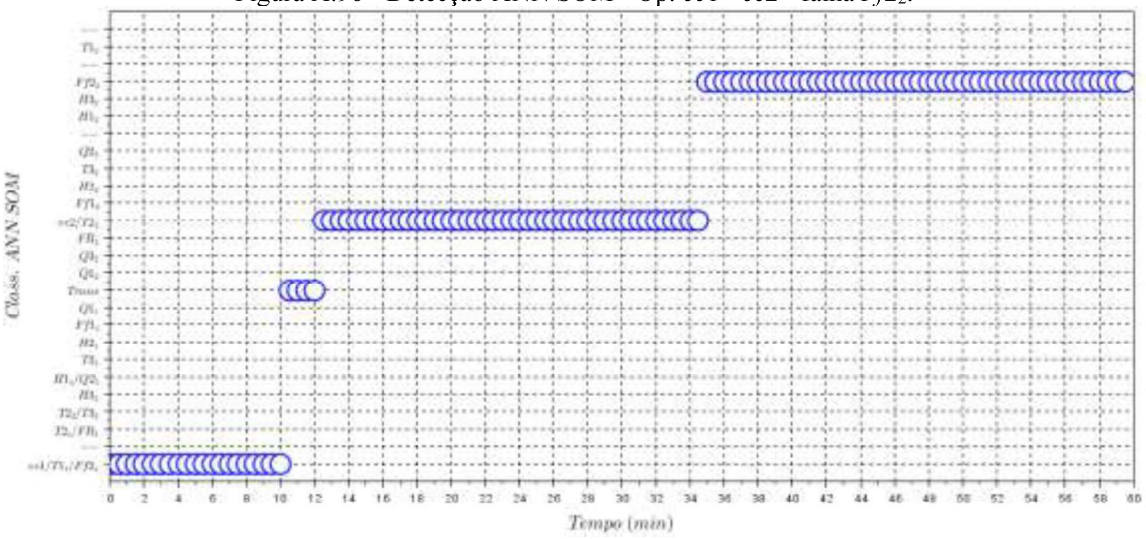


Figura A.97 - Detecção ANN SOM - Op. *ee*2 - *ee*1 - falha FR_1.

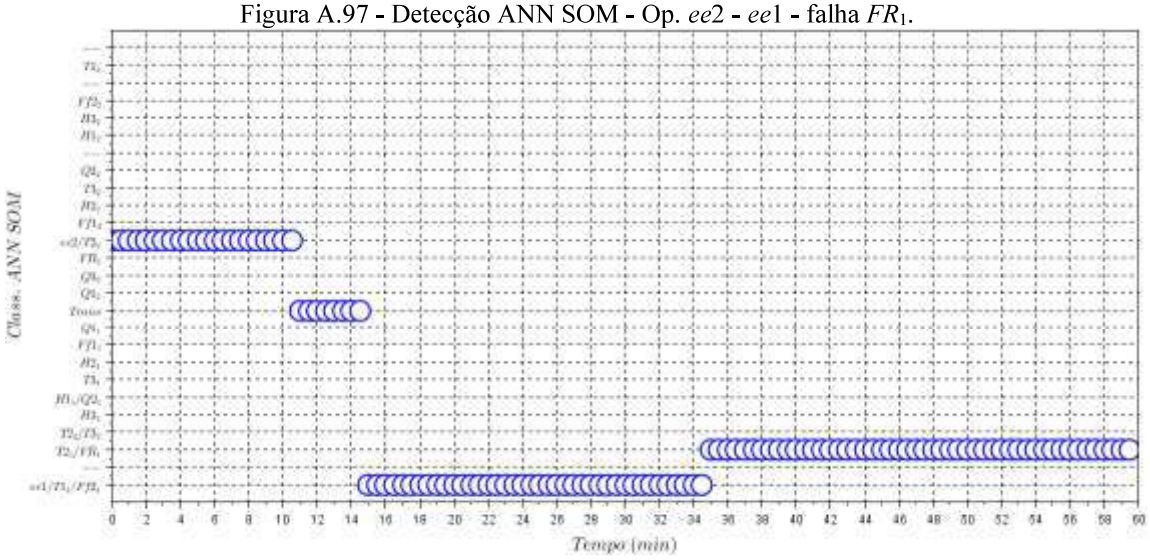


Figura A.98 - Detecção ANN SOM - Op. *ee*1 - *ee*2 - falha FR_2.

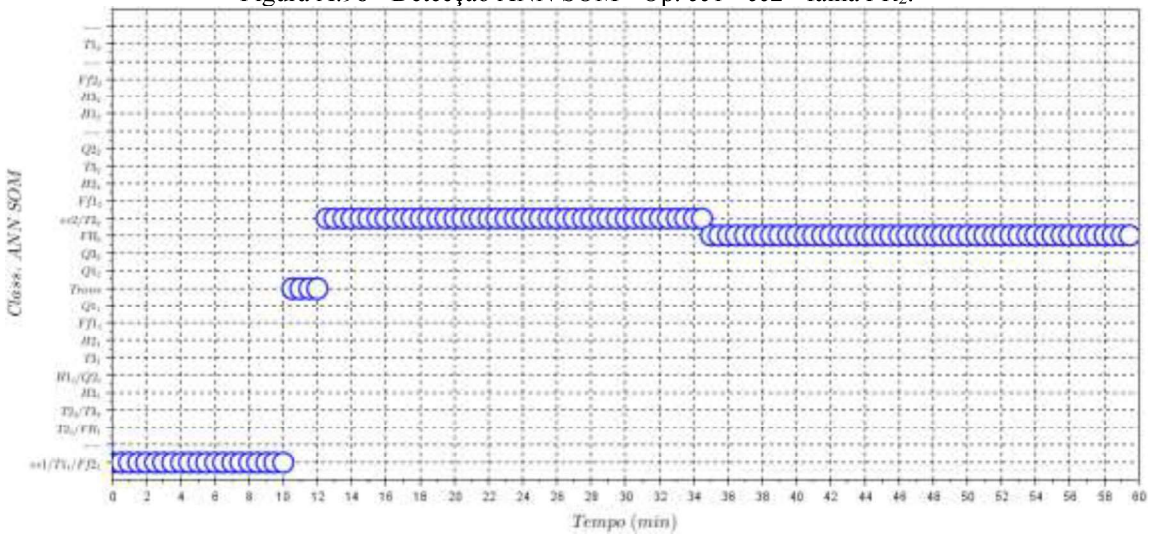


Figura A.99 - Detecção ANN SOM - Op. *ee*2 - *ee*1 - falha $Q1_1$.

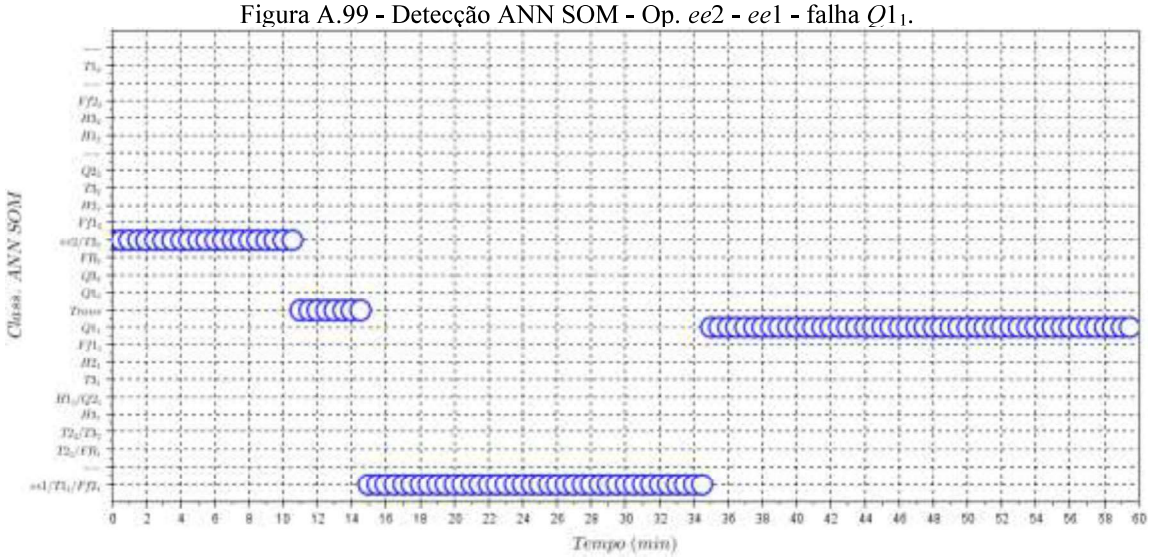


Figura A.100 - Detecção ANN SOM - Op. *ee*1 - *ee*2 - falha $Q1_2$.

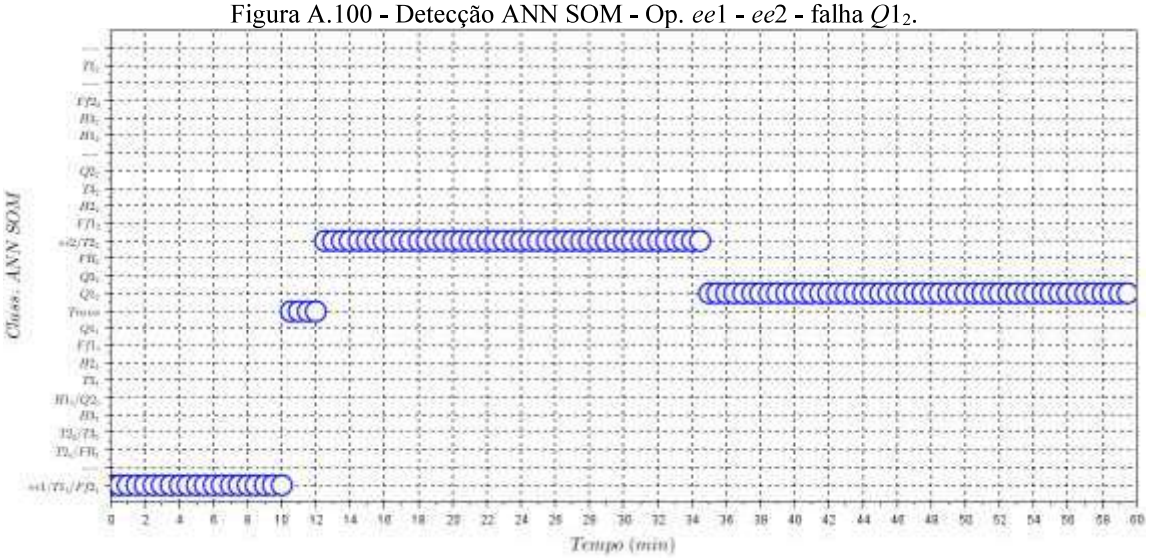


Figura A.101 - Detecção ANN SOM - Op. *ee*2 - *ee*1 - falha $Q2_1$.

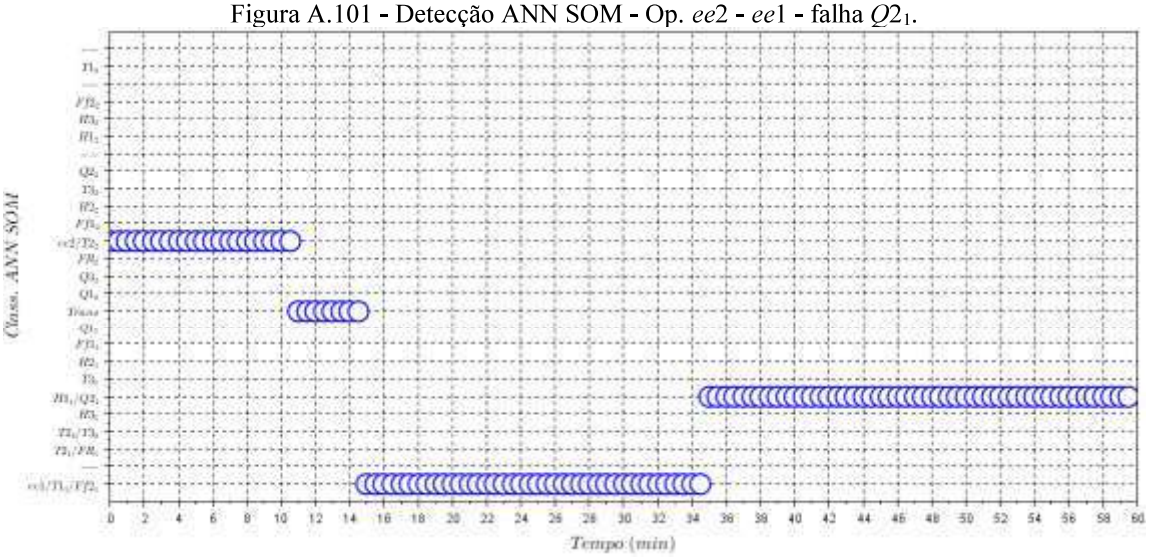


Figura A.102 - Detecção ANN SOM - Op. *ee*1 - *ee*2 - falha $Q2_2$.

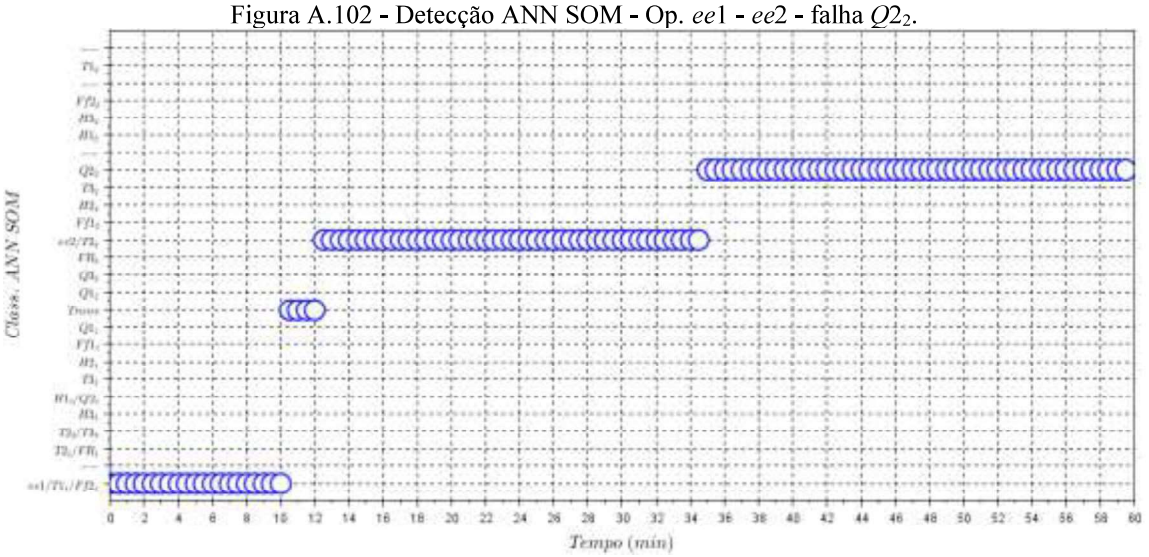


Figura A.103 - Detecção ANN SOM - Op. *ee*2 - *ee*1 - falha $Q3_1$.

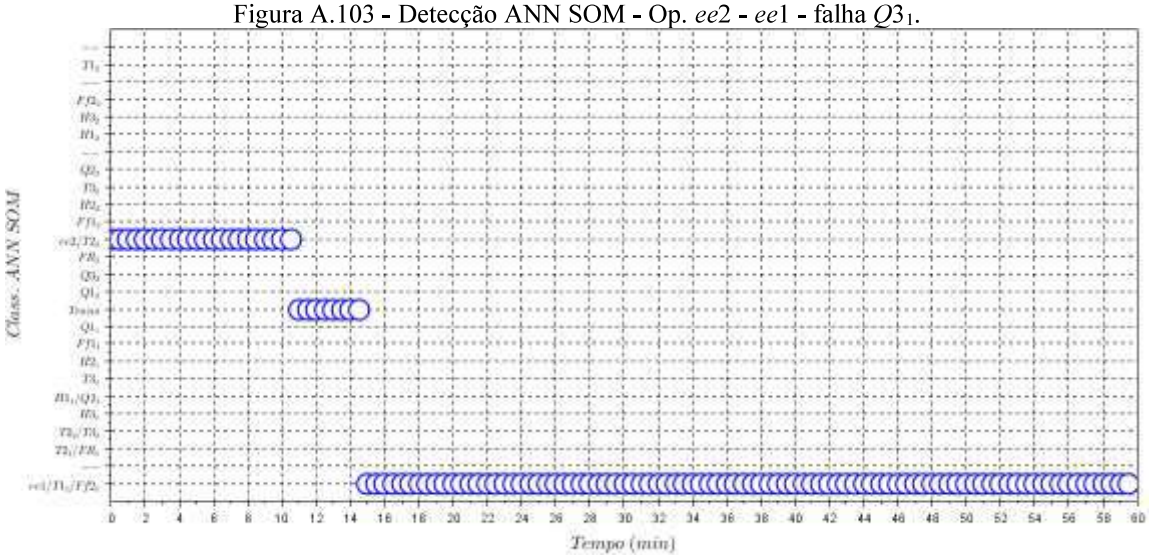
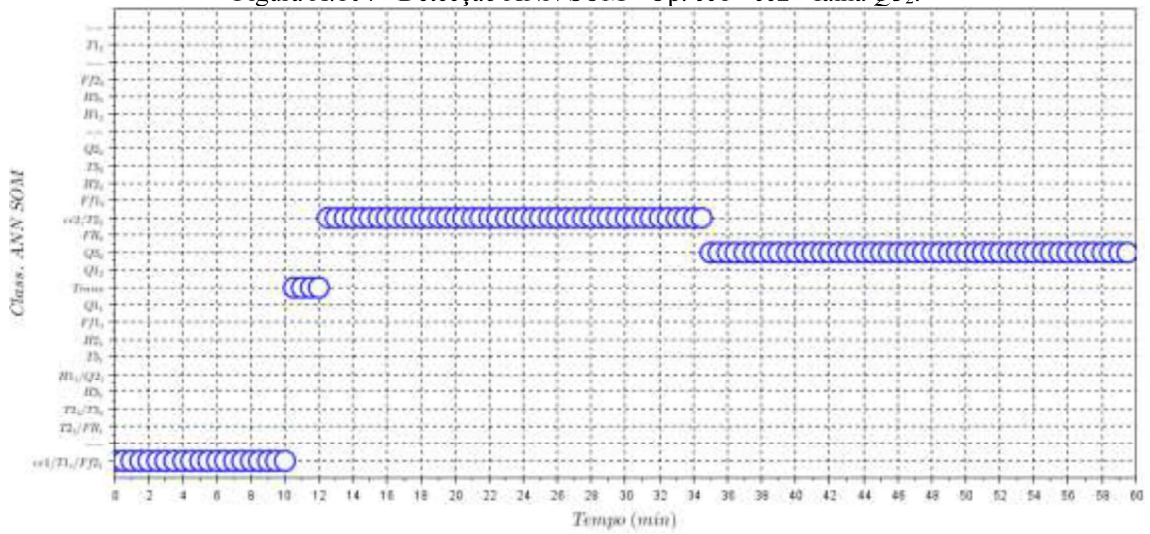


Figura A.104 - Detecção ANN SOM - Op. *ee*1 - *ee*2 - falha $Q3_2$.



Analisando a aplicação de diferentes amplitudes de falhas, pode-se afirmar que as redes neuronais artificiais, supervisionadas e não supervisionadas de classificação, são capazes de detectar falhas mesmo que não tenham sido treinadas com a mesma amplitude da falha amostrada. Nos casos em que a amplitude da falha a ser detectada é maior que a amplitude dos dados de treinamento, as metodologias são capazes de detectar mais rapidamente as diferentes falhas.

A.5. Resultados *Fuzzy*

Foram estabelecidas 24 falhas, sendo duas falhas para cada variável, aplicadas em cada um dos dois estados estacionários estipulados, de acordo com o Capítulo 3. Além das 24 falhas, foram utilizados dados de ambos os estados estacionários, totalizando 26 situações passíveis de detecção. A metodologia de detecção de falhas *Fuzzy* utiliza funções de pertinência de entrada

baseados em valores chaves dos dados disponíveis de forma a fuzzificar os valores de entrada de forma que correspondam a funções de pertinência de saída, classificando assim as diferentes falhas e estados estacionários. Foram criadas funções de pertinência para cada variável controlada e cada variável manipulada, que são consideradas como entradas do sistema de detecção. Para a saída do sistema de detecção, foram desenvolvidas funções de pertinência triangulares para cada uma das variáveis de entrada, designando se a variável se encontra no estado estacionário 1, no estado estacionário 2 ou em cada uma das falhas simuladas para as variáveis em questão.

Para todas as situações passíveis de detecção, foram simulados computacionalmente 26 conjuntos de sinais dispostos em uma matriz, em que outro estado estacionário ou falha foram aplicados nos instantes 10 minutos e 35 minutos, em um total de 60 minutos de operação, para cada sinal treinado. Os dados então foram separados em dados de treinamento e de validação. Foi utilizada uma proporção de 2 para 1 entre os dados de treinamento e validação. A cada dois instantes separados para treinamento, o terceiro instante era designado para validação.

Os resultados para cada uma das variáveis e suas falhas são apresentados nas Figuras, assim como as funções de pertinência para cada variável, de entrada e de saída.

Figura A.105 - Funções de pertinência *Fuzzy* de entrada - variável Ff_1.

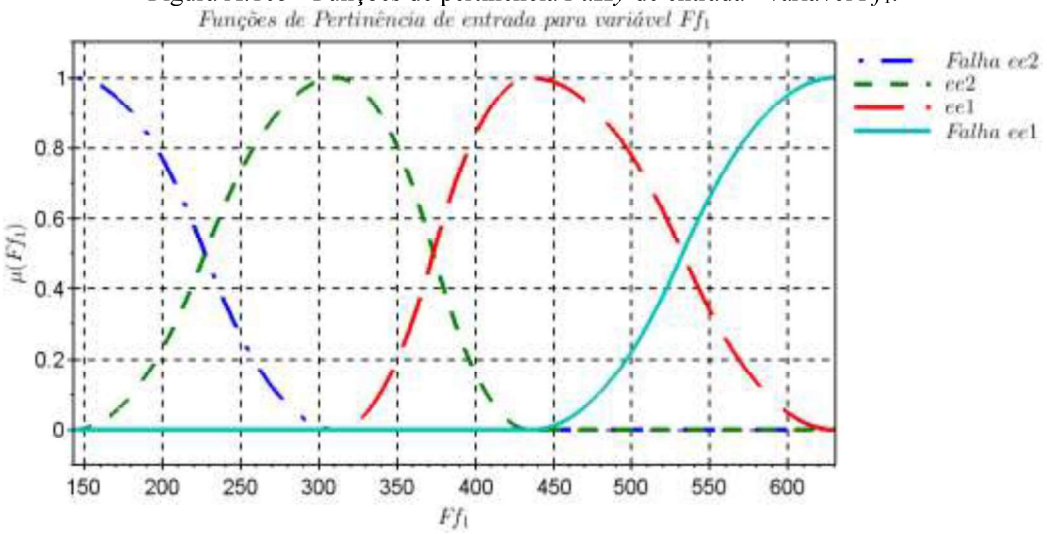


Figura A.106 - Detecção Fuzzy - Op. *ee*2 - *ee*1 - falha $Ff1_1$.

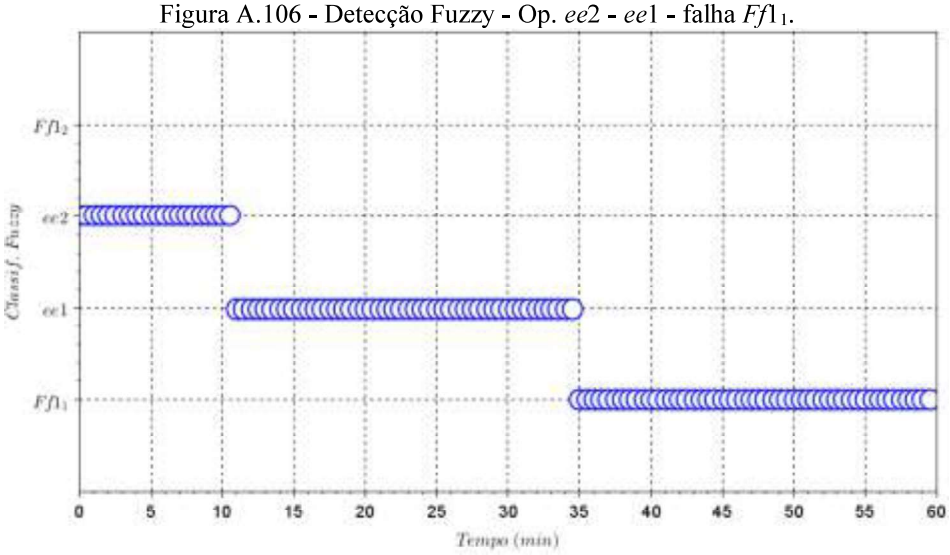


Figura A.107 - Detecção Fuzzy - Op. *ee*1 - *ee*2 - falha $Ff1_2$.

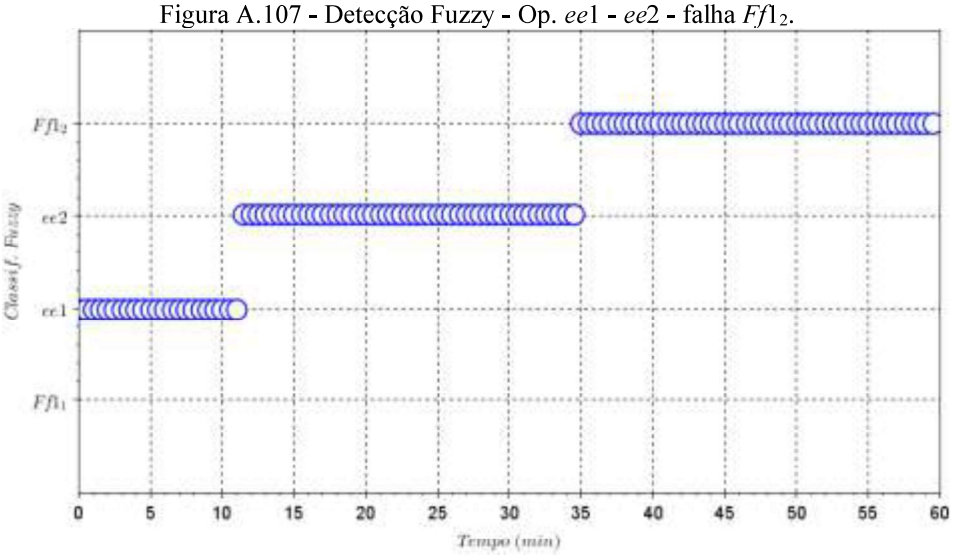


Figura A.108 - Funções de pertinência *Fuzzy* de entrada - variável Ff_2.

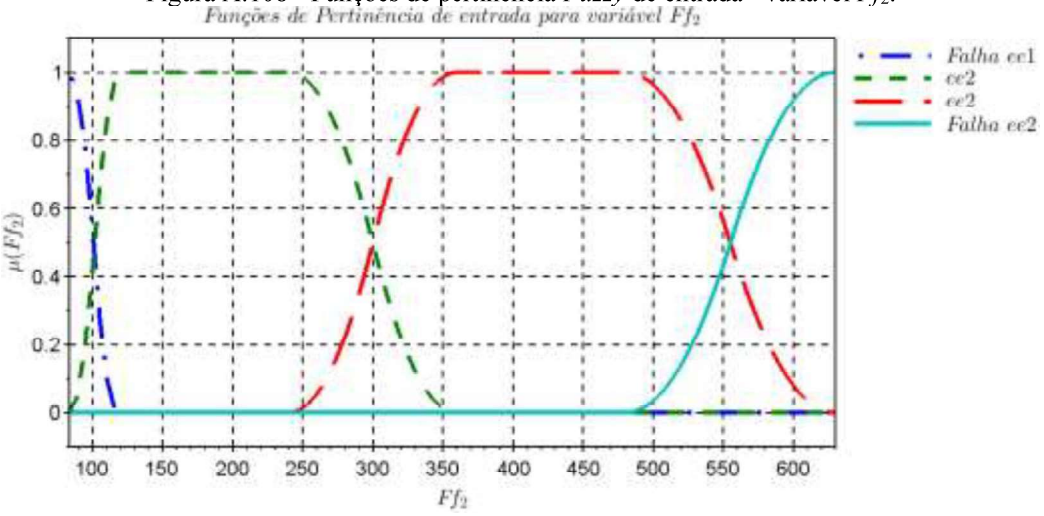


Figura A.109 - Detecção Fuzzy - Op. *ee*2 - *ee*1 - falha $Ff2_1$.

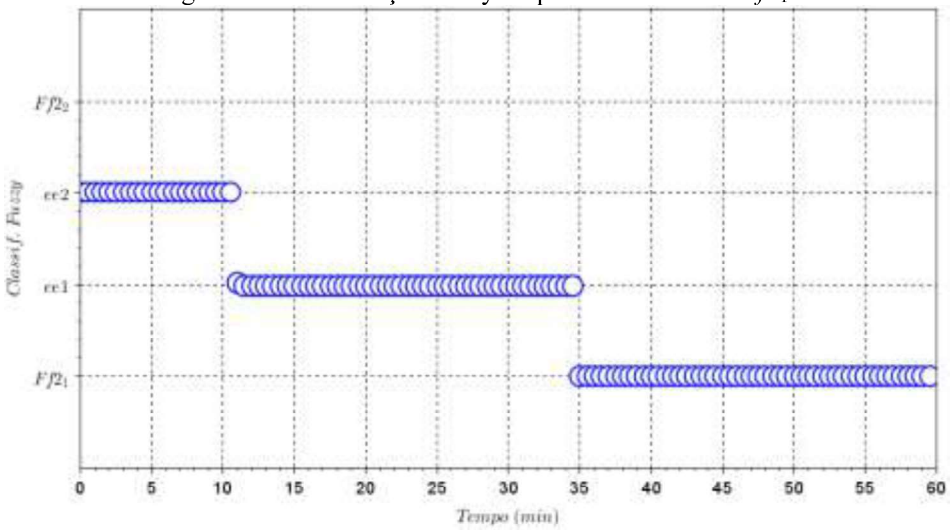


Figura A.110 - Detecção Fuzzy - Op. *ee*1 - *ee*2 - falha $Ff2_2$.

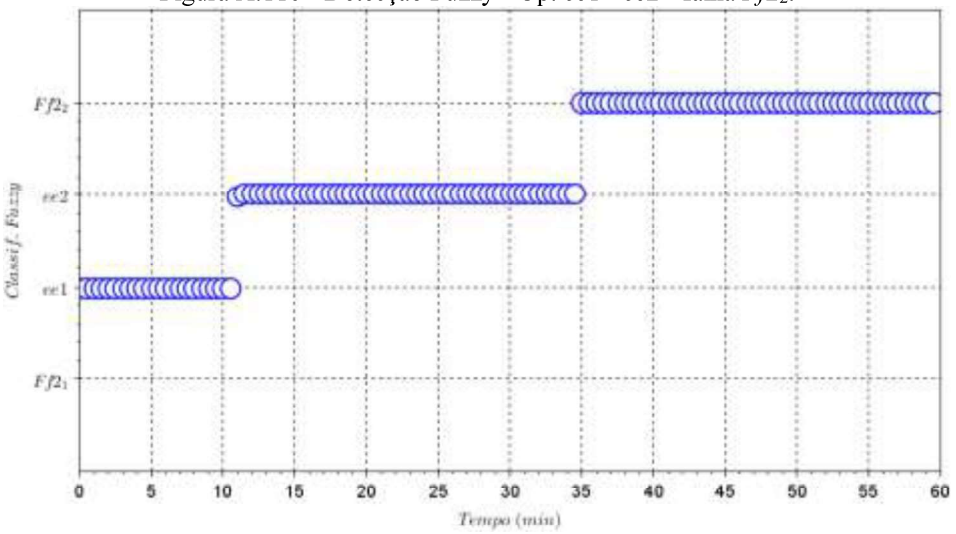


Figura A.111 - Funções de pertinência *Fuzzy* de entrada - variável F_R.

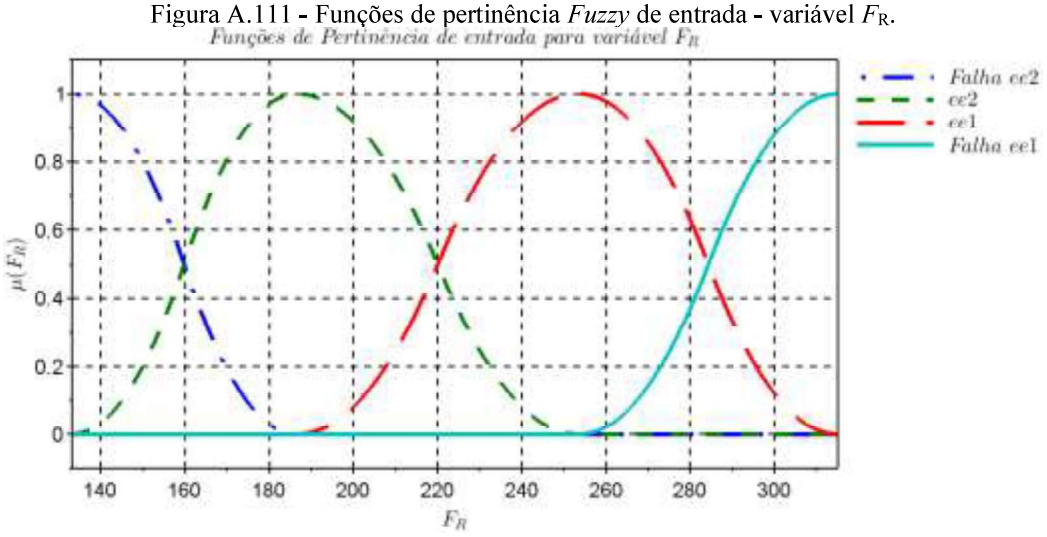


Figura A.112 - Detecção Fuzzy - Op. *ee*2 - *ee*1 - falha F_{R1}.

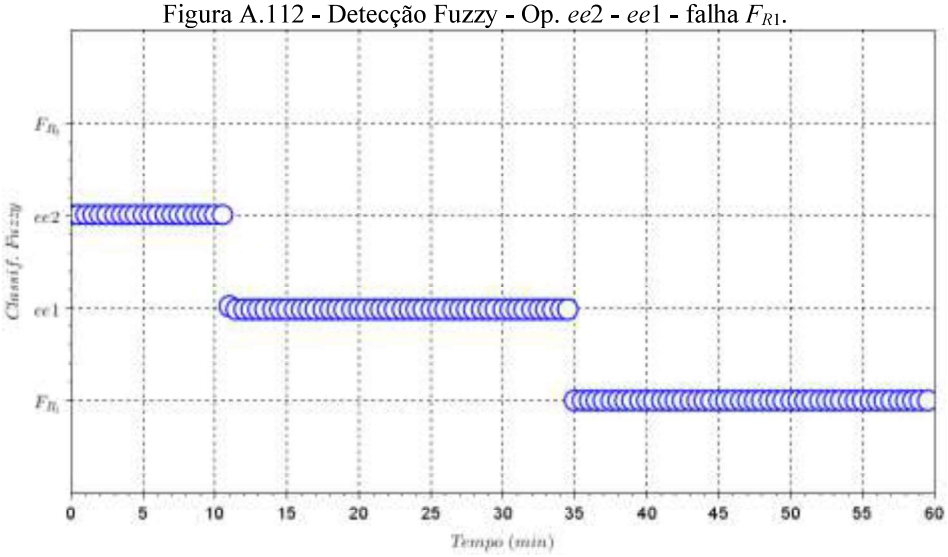


Figura A.113 - Detecção Fuzzy - Op. *ee*1 - *ee*2 - falha F_{R2}.

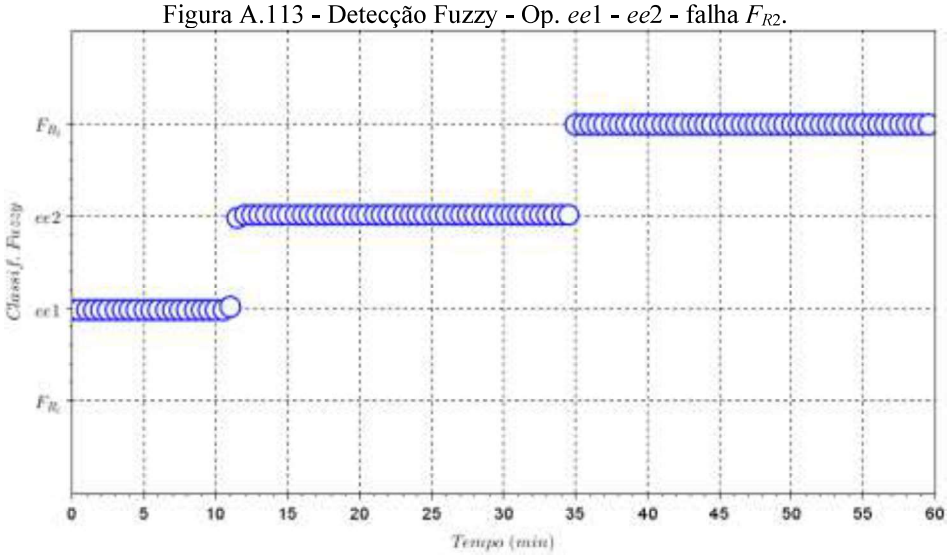


Figura A.114 - Funções de pertinência *Fuzzy* de entrada - variável H_1.

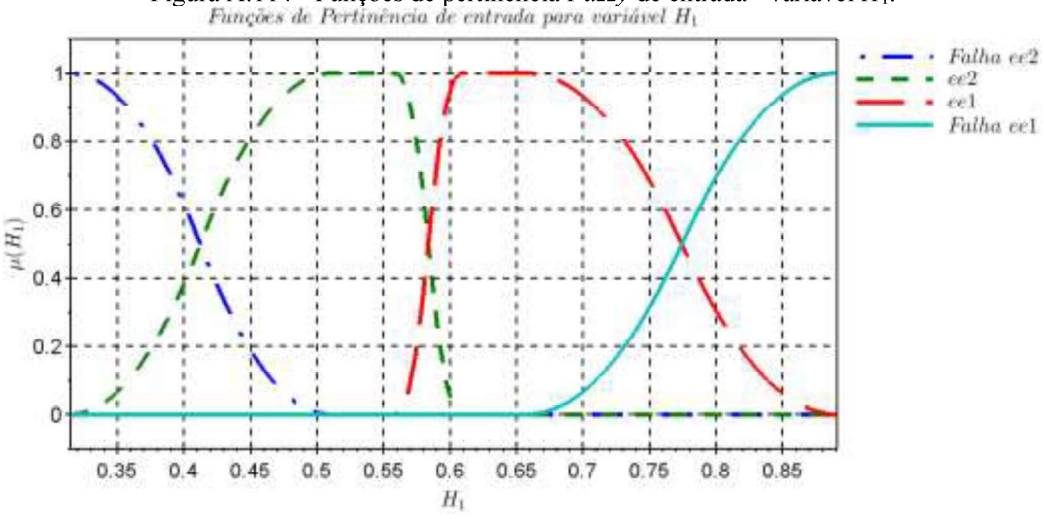


Figura A.115 - Detecção Fuzzy - Op. *ee*2 - *ee*1 - falha $H1_1$.

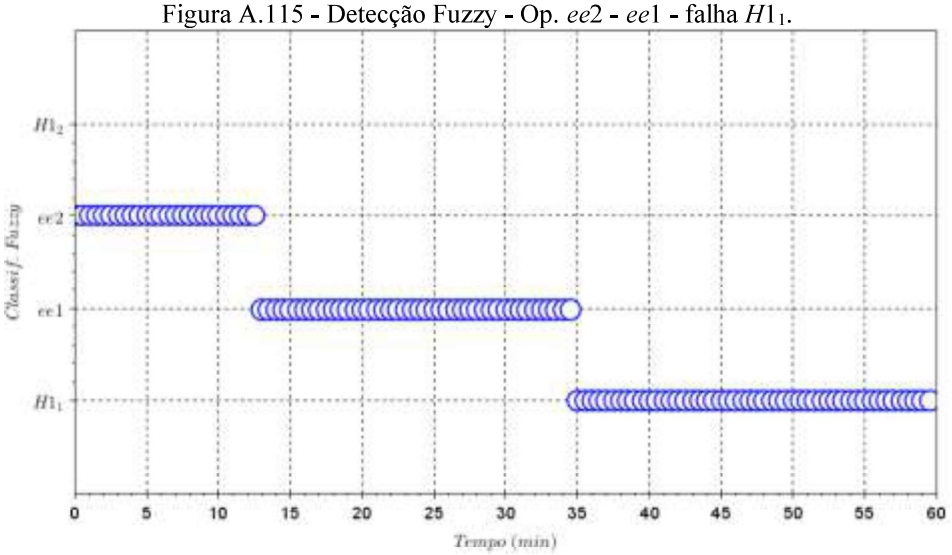


Figura A.116 - Detecção Fuzzy - Op. *ee*1 - *ee*2 - falha $H1_2$.

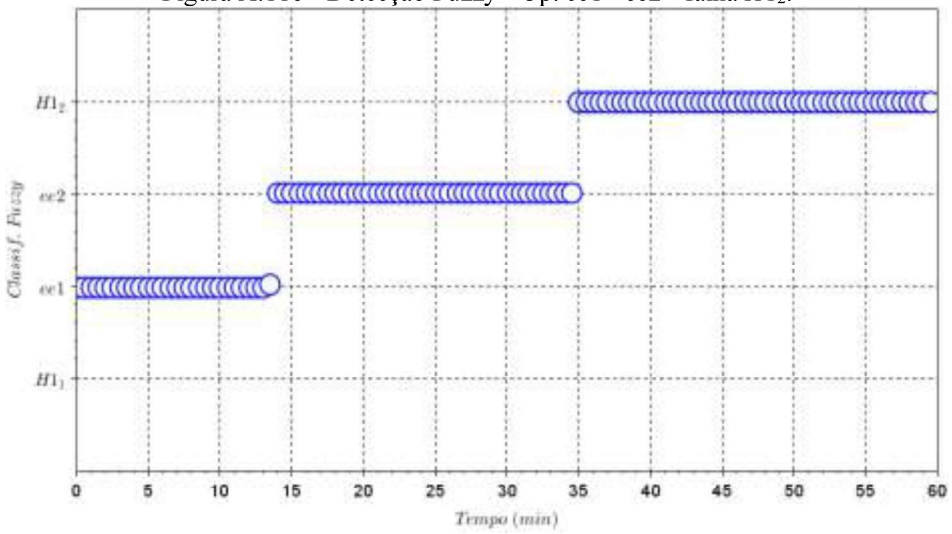


Figura A.117 - Funções de pertinência *Fuzzy* de entrada - variável H_2.

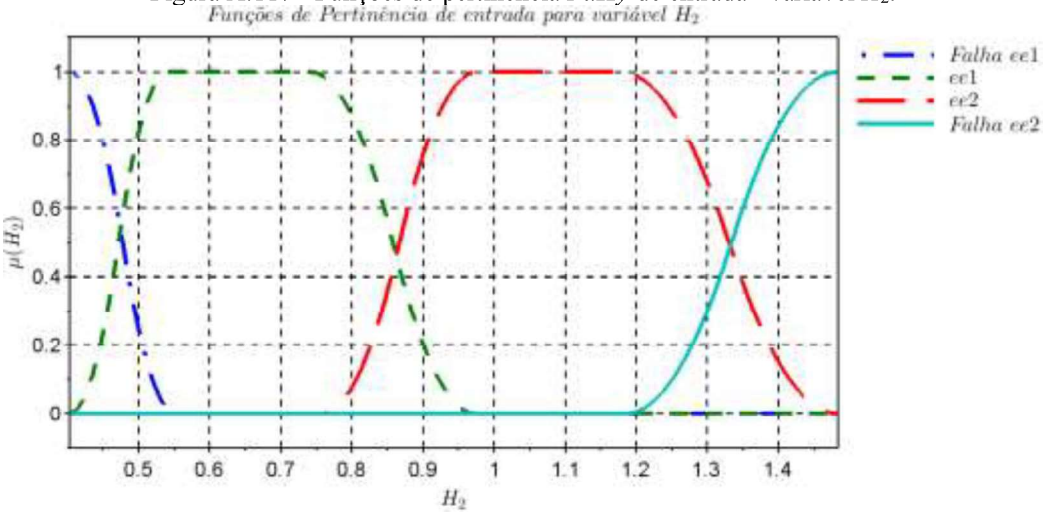


Figura A.118 - Detecção Fuzzy - Op. *ee*2 - *ee*1 - falha $H2_1$.

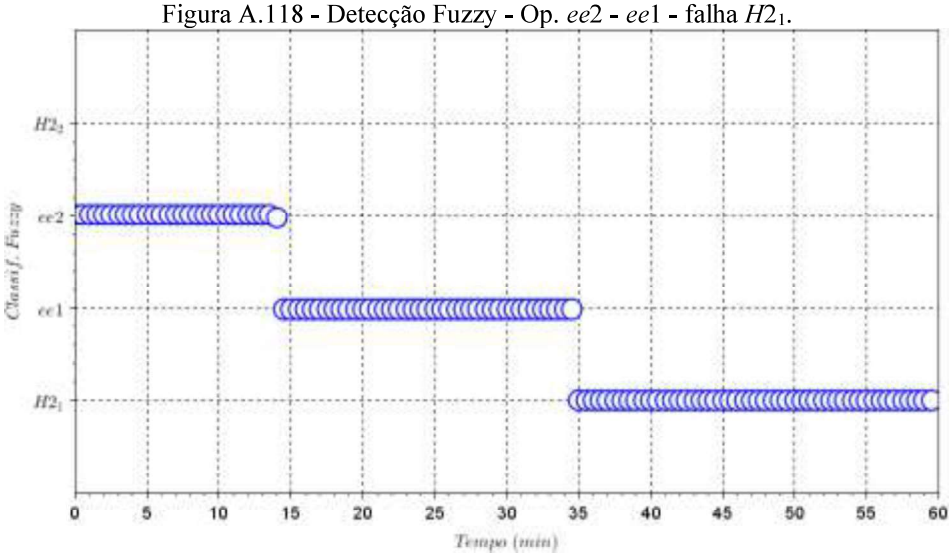


Figura A.119 - Detecção Fuzzy - Op. *ee*1 - *ee*2 - falha $H2_2$.

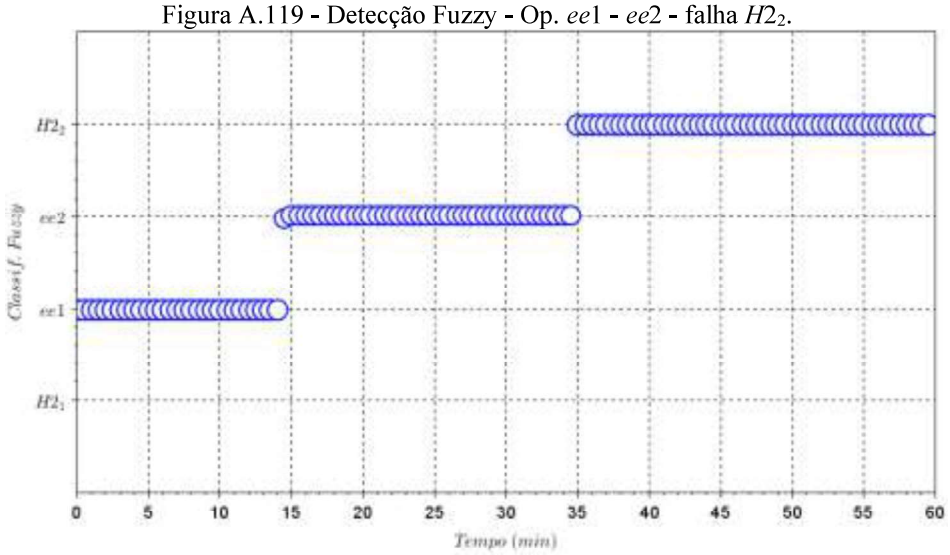


Figura A.120 - Funções de pertinência *Fuzzy* de entrada - variável H_3.

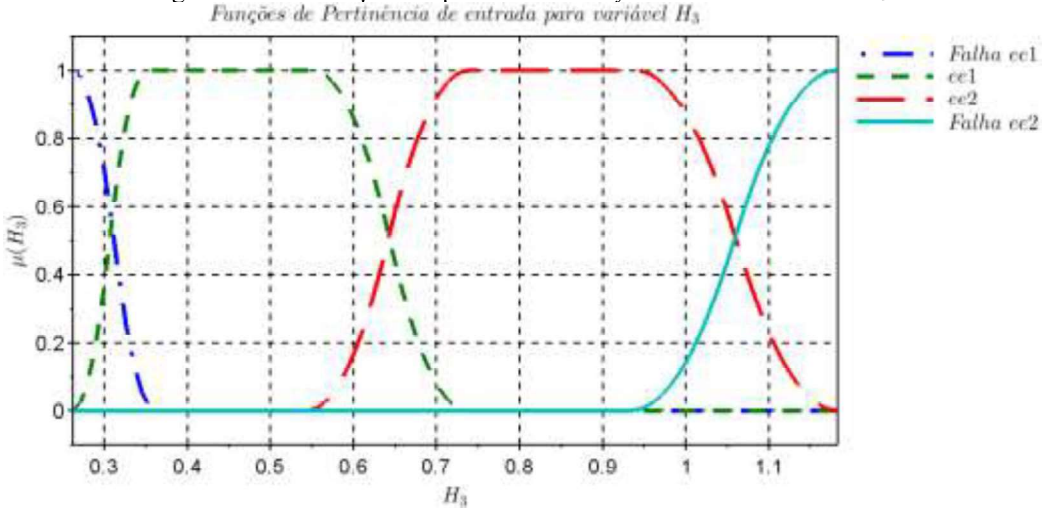


Figura A.121 - Detecção Fuzzy - Op. *ee*2 - *ee*1 - falha $H3_1$.

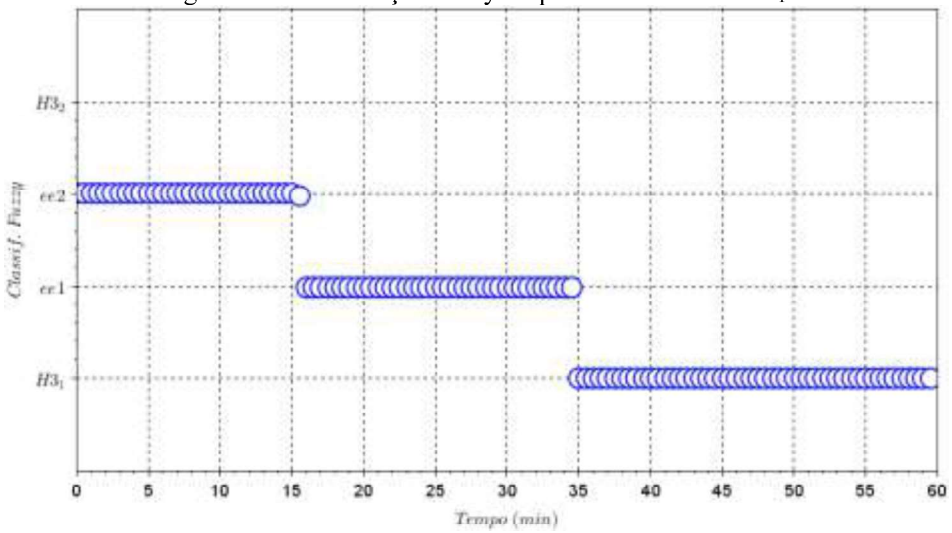


Figura A.122 - Detecção Fuzzy - Op. *ee*1 - *ee*2 - falha $H3_2$.

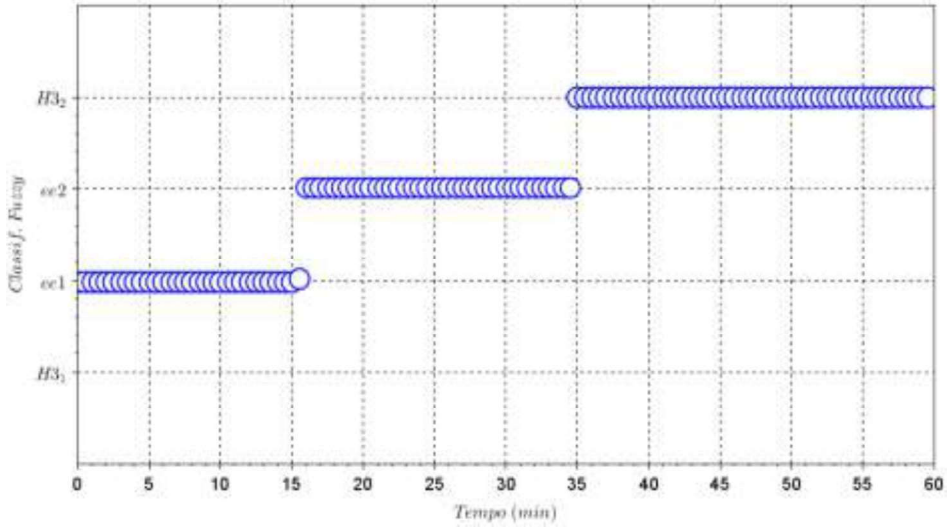


Figura A.123 - Funções de pertinência *Fuzzy* de entrada - variável Q_1.

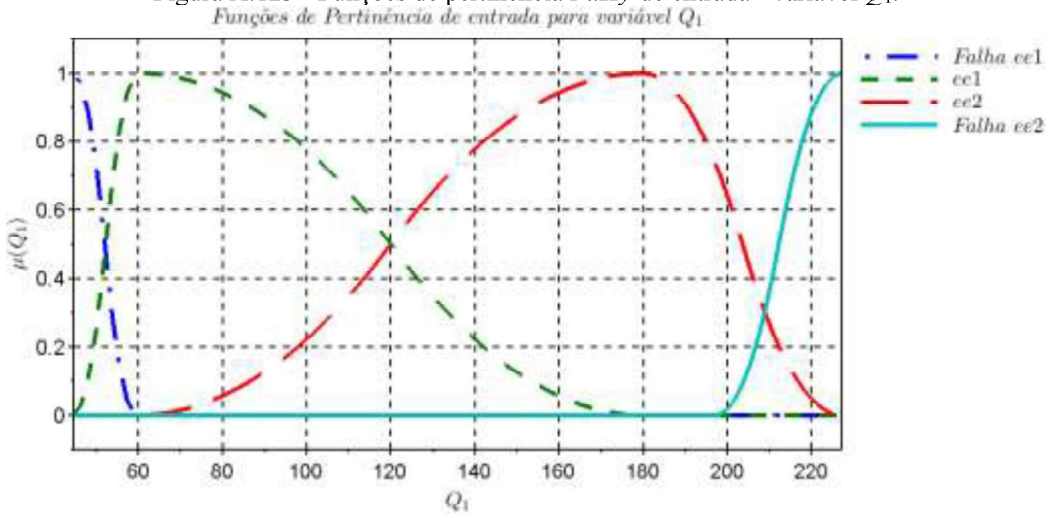


Figura A.124 - Detecção Fuzzy - Op. *ee*2 - *ee*1 - falha $Q1_1$.

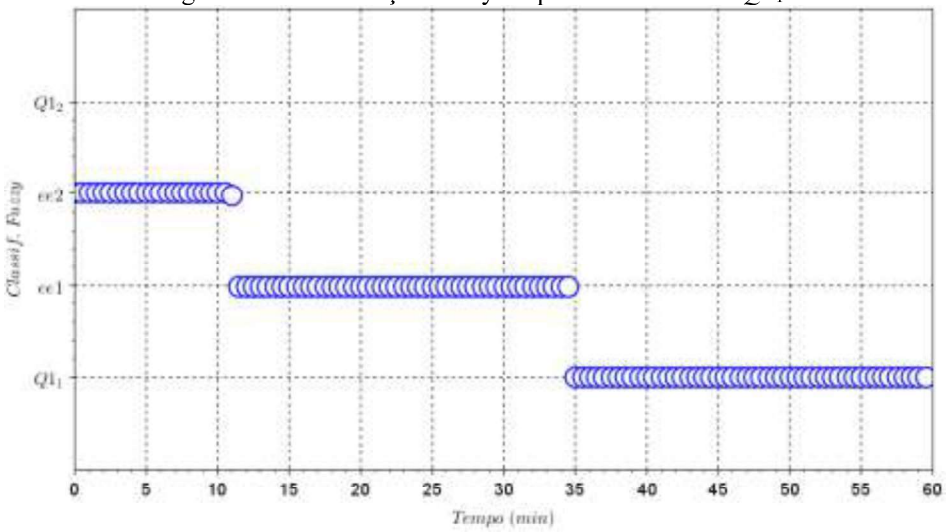


Figura A.125 - Detecção Fuzzy - Op. *ee*1 - *ee*2 - falha $Q1_2$.

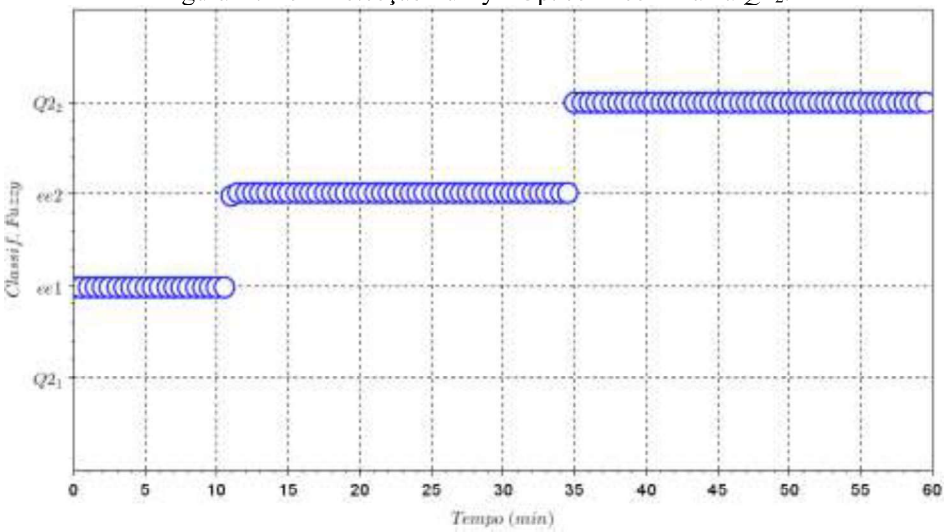


Figura A.126 - Funções de pertinência *Fuzzy* de entrada - variável Q_2.

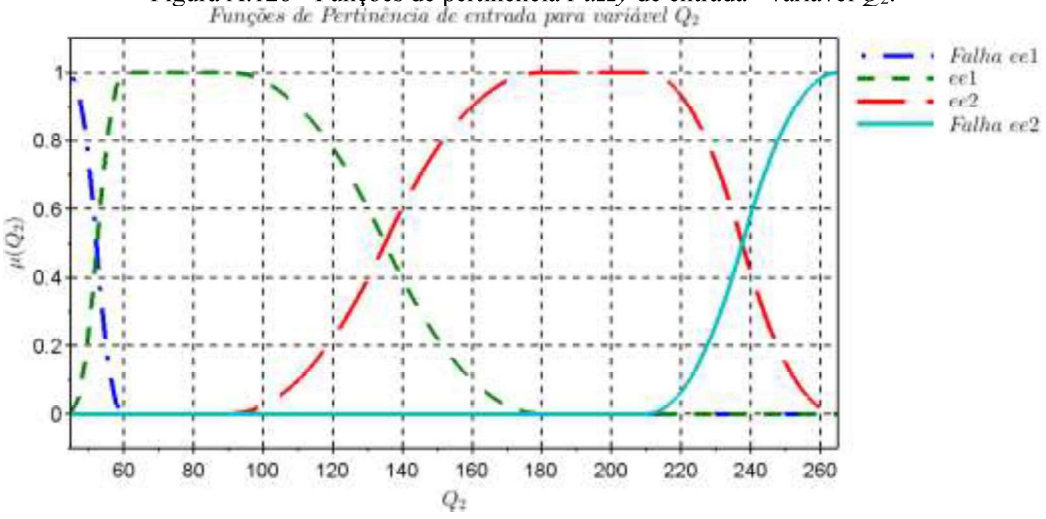


Figura A.127 - Detecção Fuzzy - Op. *ee*2 - *ee*1 - falha $Q2_1$.

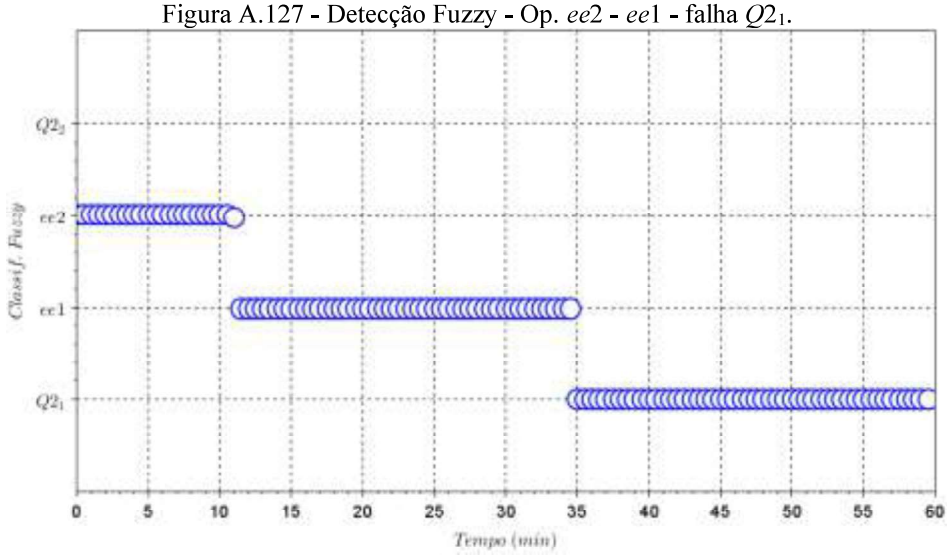


Figura A.128 - Detecção Fuzzy - Op. *ee*1 - *ee*2 - falha $Q2_2$.

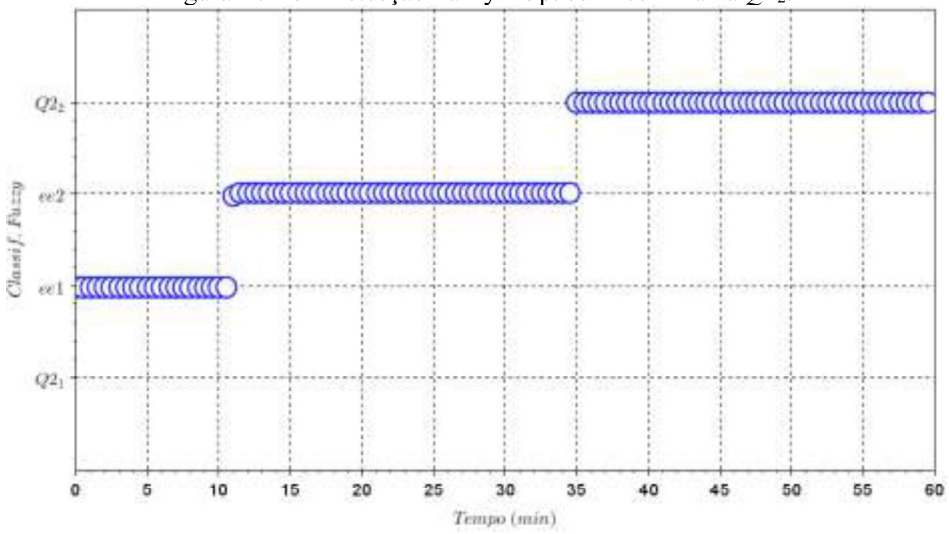


Figura A.129 - Funções de pertinência *Fuzzy* de entrada - variável Q_3.

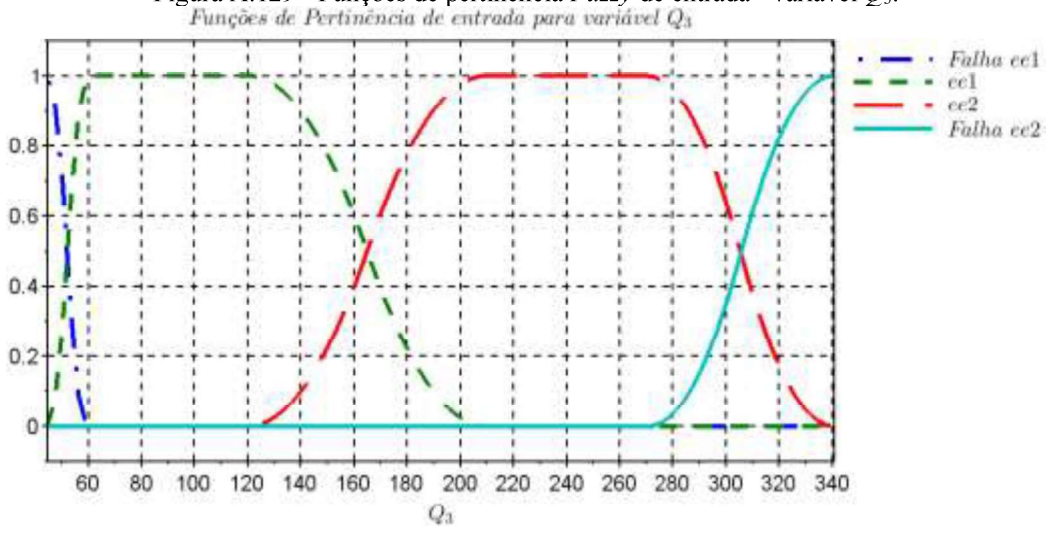


Figura A.130 - Detecção Fuzzy - Op. *ee*2 - *ee*1 - falha $Q3_1$.

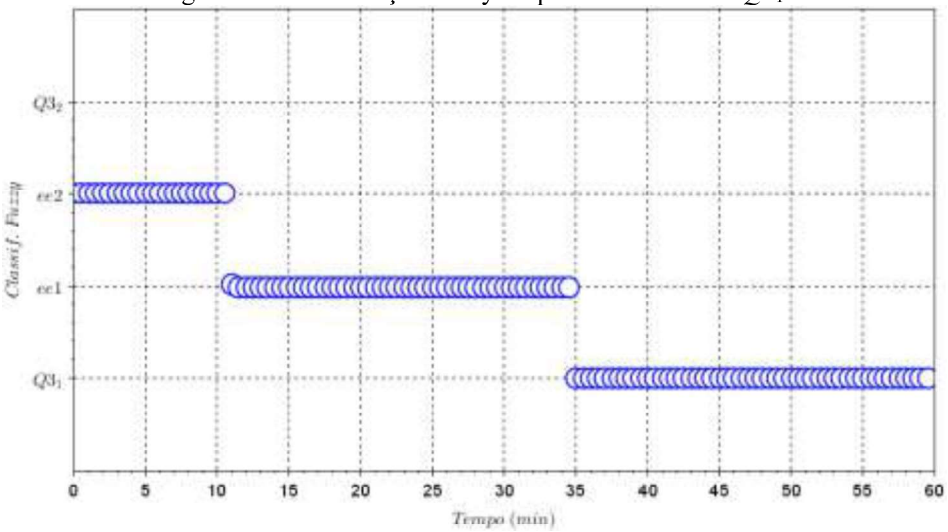


Figura A.131 - Detecção Fuzzy - Op. *ee*1 - *ee*2 - falha $Q3_2$.

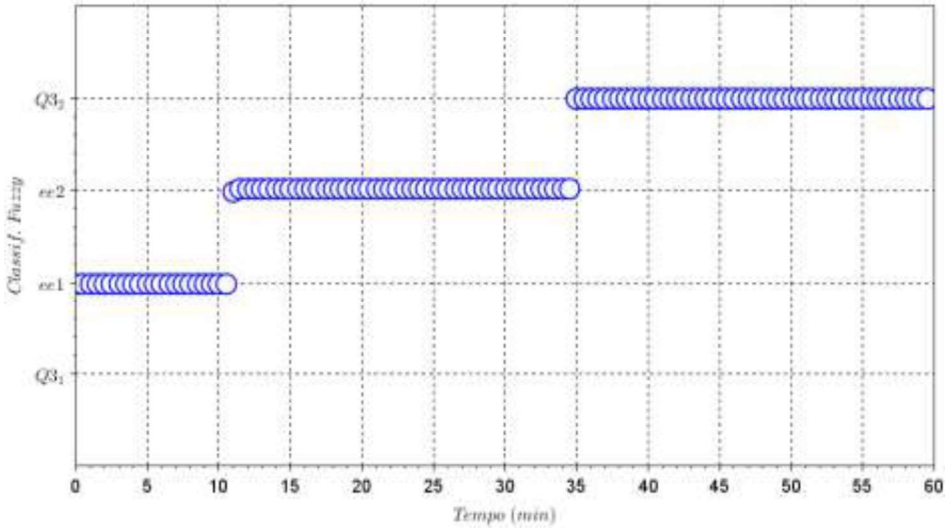


Figura A.132 - Funções de pertinência *Fuzzy* de entrada - variável T_1.

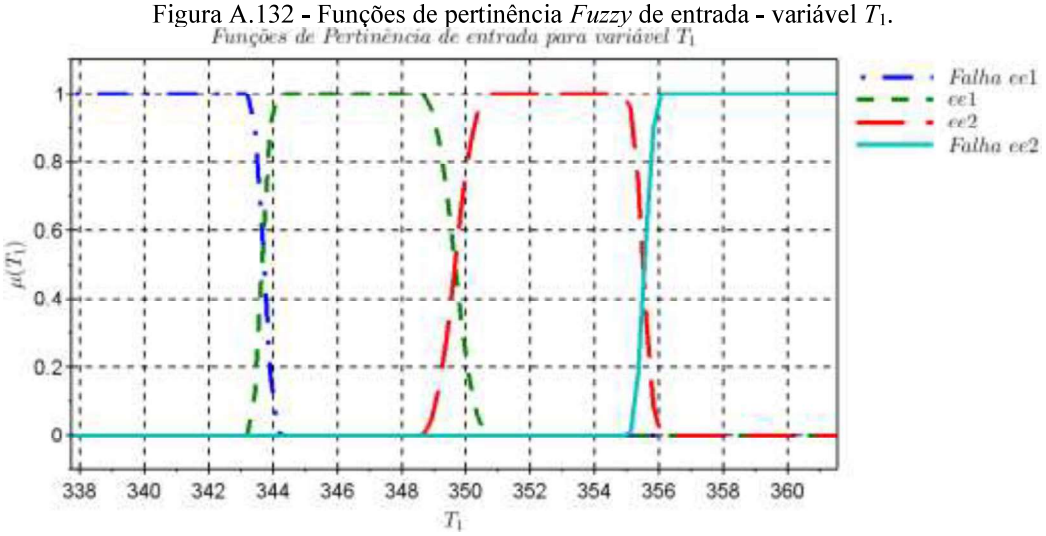


Figura A.133 - Detecção Fuzzy - Op. *ee*2 - *ee*1 - falha $T1_1$.

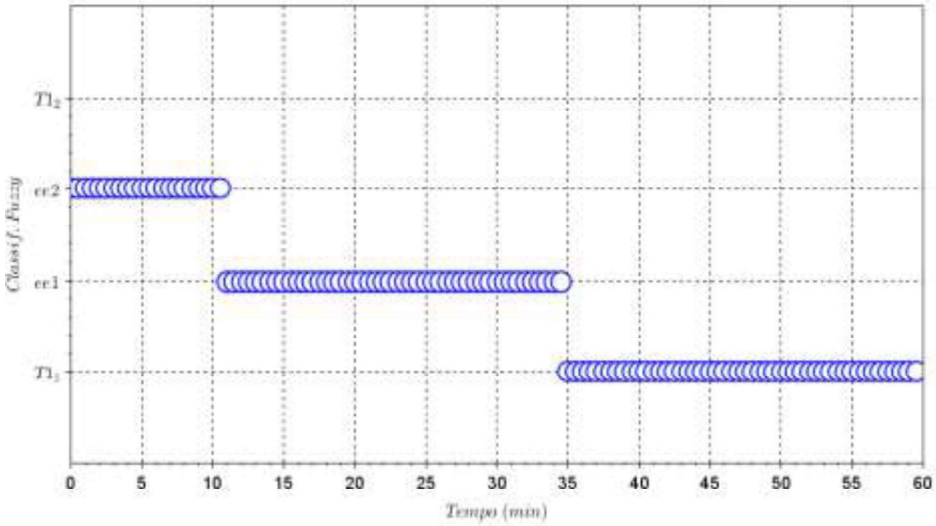


Figura A.134 - Detecção Fuzzy - Op. *ee*1 - *ee*2 - falha $T1_2$.

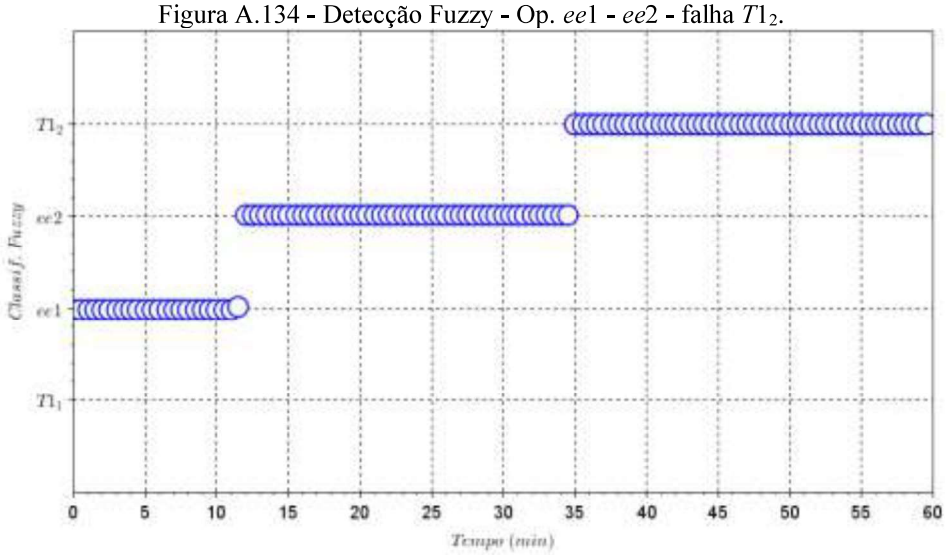


Figura A.135 - Funções de pertinência *Fuzzy* de entrada - variável T_2.

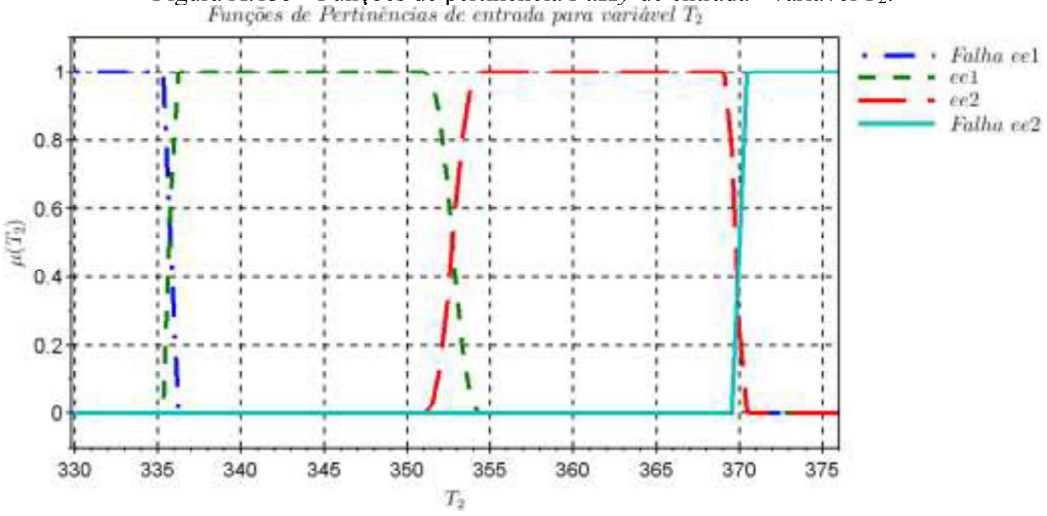


Figura A.136 - Detecção Fuzzy - Op. *ee*2 - *ee*1 - falha $T2_1$.

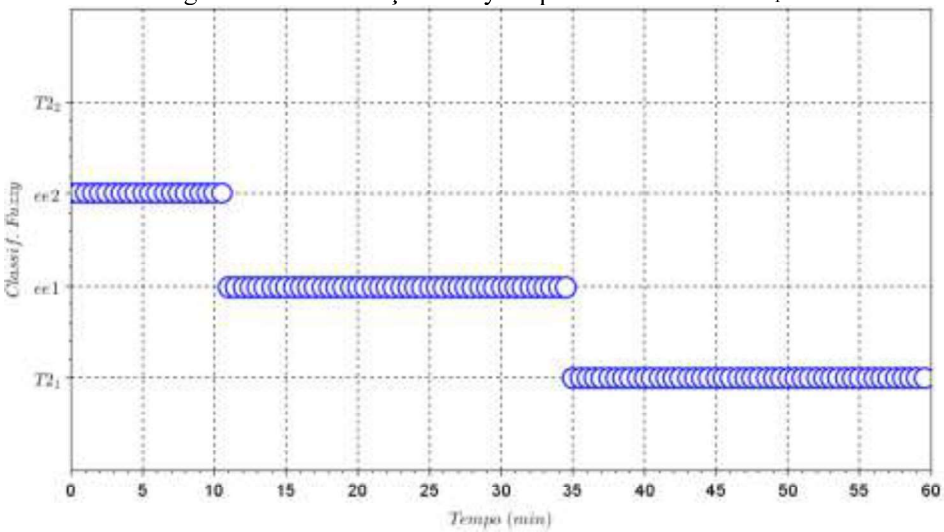


Figura A.137 - Detecção Fuzzy - Op. *ee*1 - *ee*2 - falha $T2_2$.

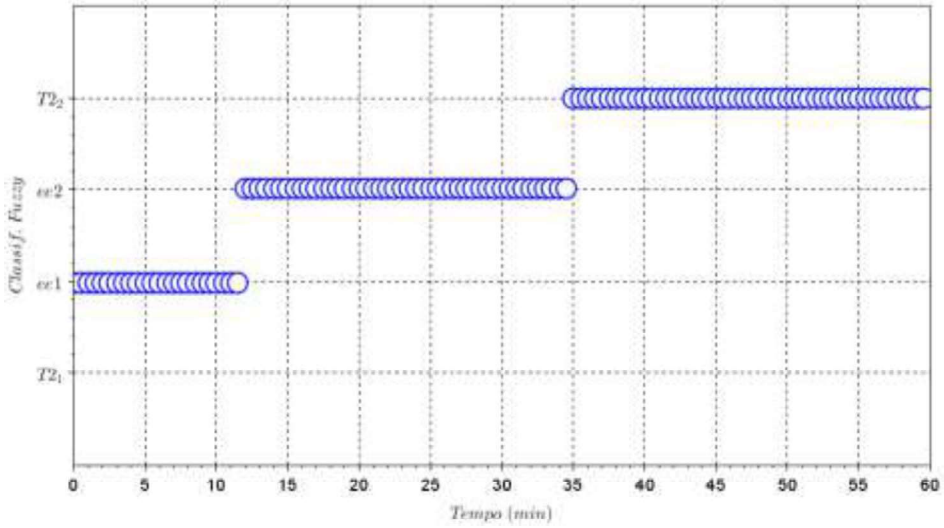


Figura A.138 - Funções de pertinência *Fuzzy* de entrada - variável T_3.

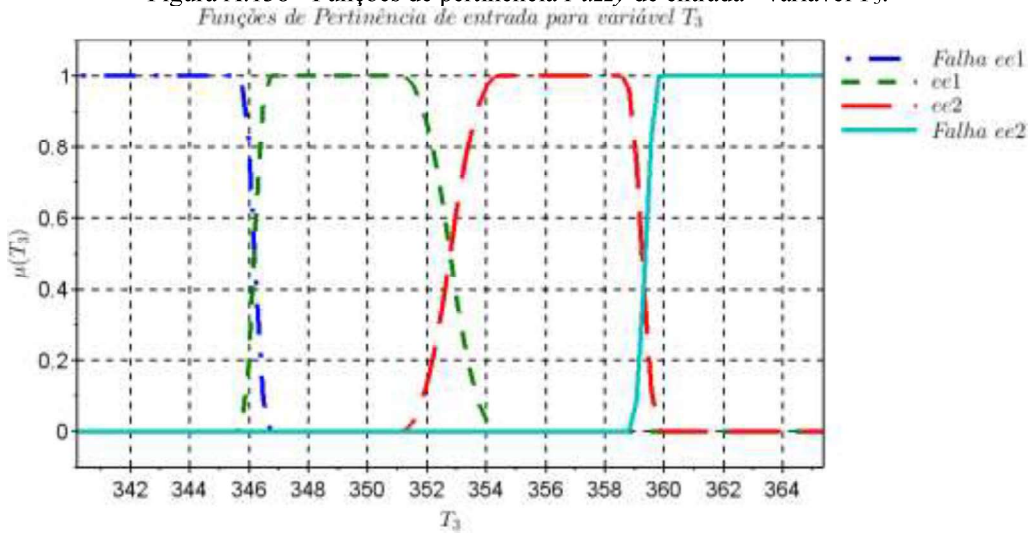


Figura A.139 - Detecção Fuzzy - Op. *ee*2 - *ee*1 - falha $T3_1$.

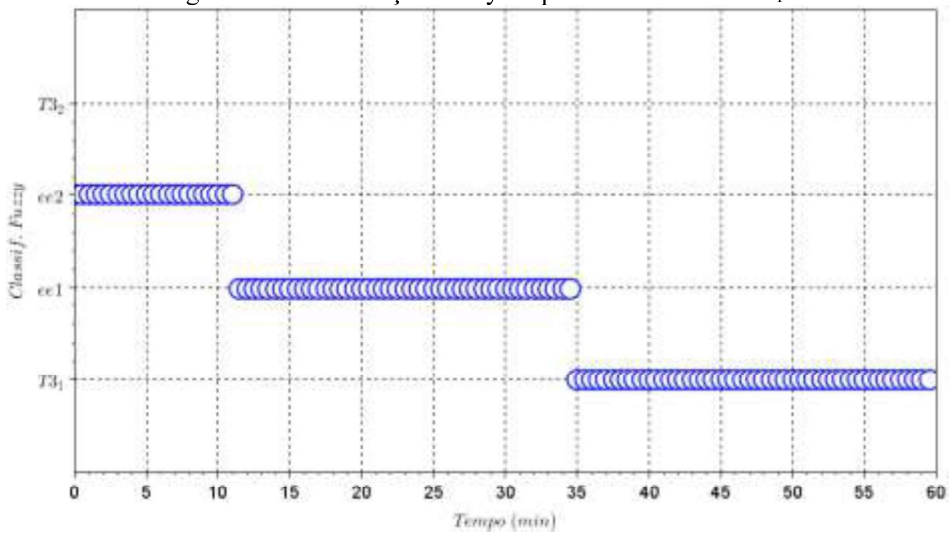


Figura A.140 - Detecção Fuzzy - Op. *ee*1 - *ee*2 - falha $T3_2$.

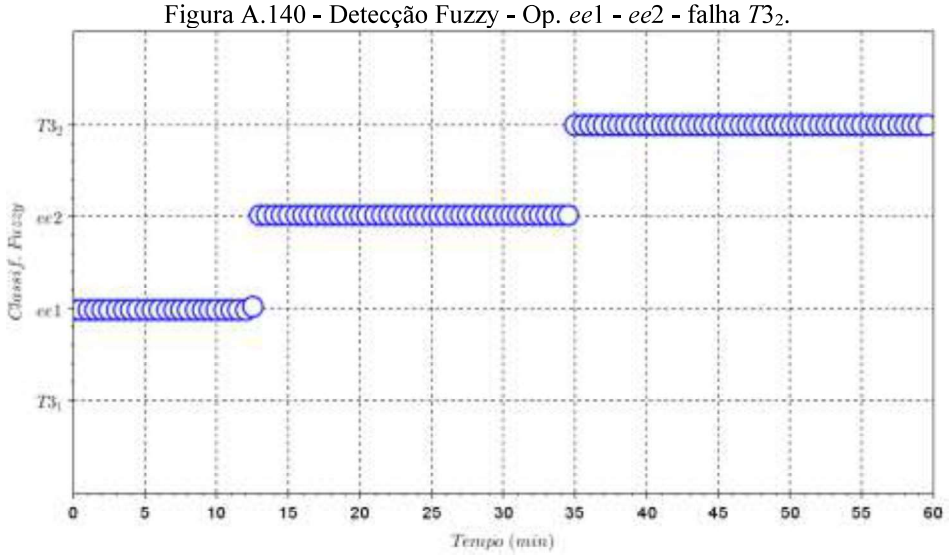
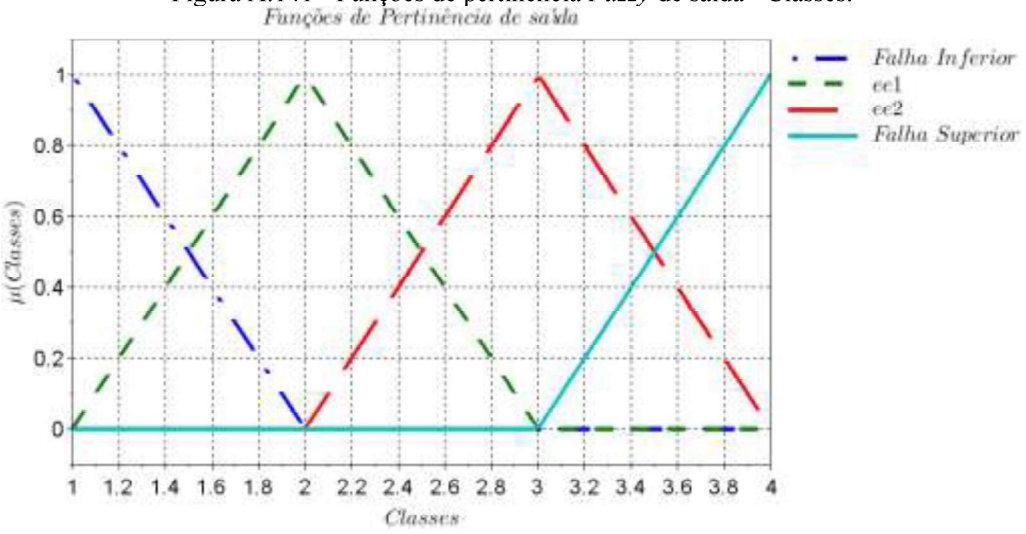


Figura A.141 - Funções de pertinência *Fuzzy* de saída - Classes.



A Lógica *Fuzzy* quando utilizada como metodologia de detecção de falhas possui vantagens perante outras metodologias como não necessitar de um procedimento de treinamento como os métodos utilizando redes neuronais e máquinas de vetores de suporte, além de ser capaz de detectar as falhas de maneira contínua, não apresentando descontinuidades nos dados de detecção, entretanto, existe uma dificuldade na obtenção das melhores funções de pertinência para determinado sistema de detecção de falhas. Os resultados obtidos através da Lógica *Fuzzy* foram compatíveis com os outros métodos de detecção.

A.6. Resultados SNN

Foram estabelecidas 24 falhas, sendo duas falhas para cada variável, aplicadas em cada um dos dois estados estacionários estipulados, de acordo com o Capítulo 3. Além das 24 falhas, foram utilizados dados de ambos os estados estacionários, totalizando 26 situações passíveis de detecção. A metodologia de detecção de falhas SNN utiliza os valores de entrada (variáveis manipuladas e variáveis controladas) de forma que correspondam as probabilidades de cada conjunto de dados fazer ou não parte de uma determinada classe, assim como as redes neuronais. A diferença se encontra no número de camadas ocultas da rede, que no caso da SNN é somente uma única camada. Para a rede SNN foi estabelecida uma camada com o dobro de nós da camada de saída, ou seja, 52 nós. Foram estabelecidas 1000 (mil) épocas para o treinamento da rede SNN.

Para todas as situações passíveis de detecção, foram simulados computacionalmente 26 conjuntos de sinais dispostos em uma matriz, em que outro estado estacionário ou falha foram aplicados nos instantes 10 minutos e 35 minutos, em um total de 60 minutos de operação, para cada sinal treinado. Os dados então foram separados em dados de treinamento e de validação. Foi utilizada uma proporção de 2 para 1 entre os dados de treinamento e validação. A cada dois instantes separados para treinamento, o terceiro instante era designado para validação. Os resultados da detecção para cada uma das variáveis e suas falhas são apresentados nas figuras a seguir.

Figura A.142 - Detecção SNN - Op. *ee*2 - *ee*1 - *ee*2.

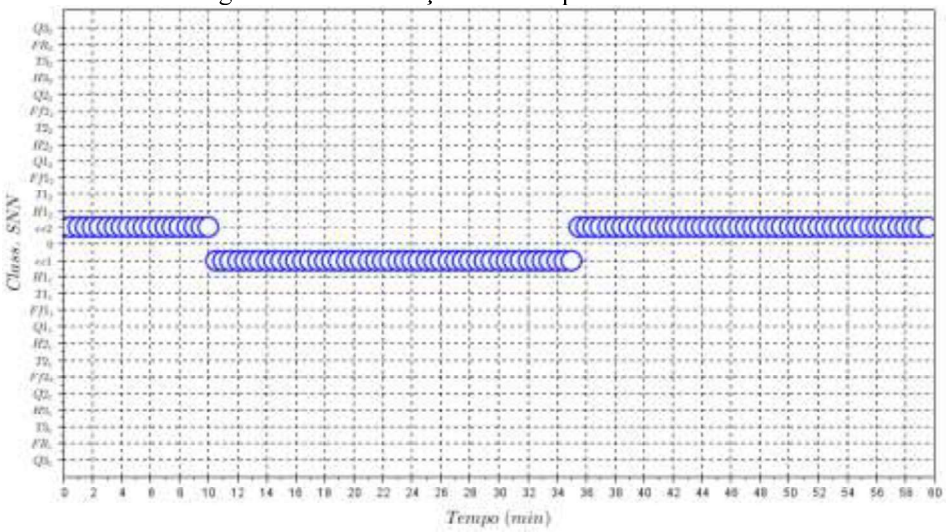


Figura A.143 - Detecção SNN - Op. *ee*1 - *ee*2 - *ee*1.

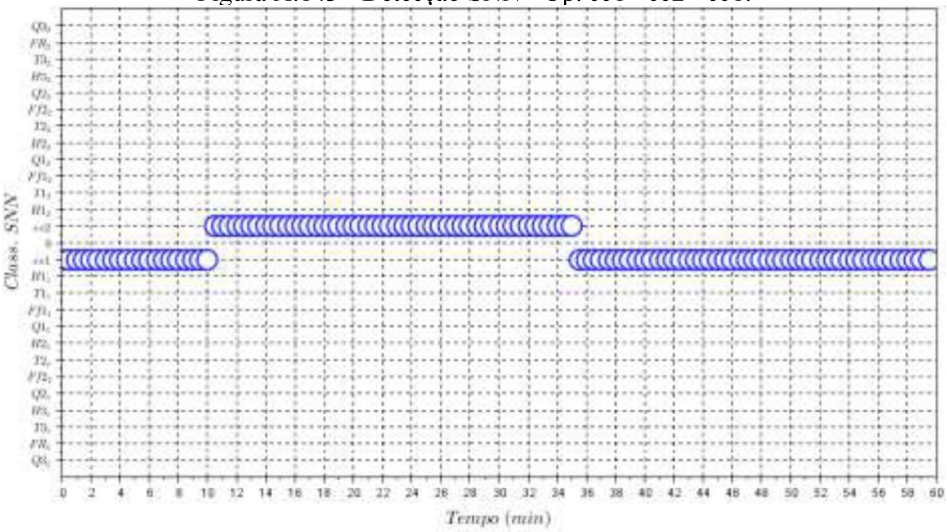


Figura A.144 - Detecção SNN - Op. *ee*2 - *ee*1 - $H1_1$.

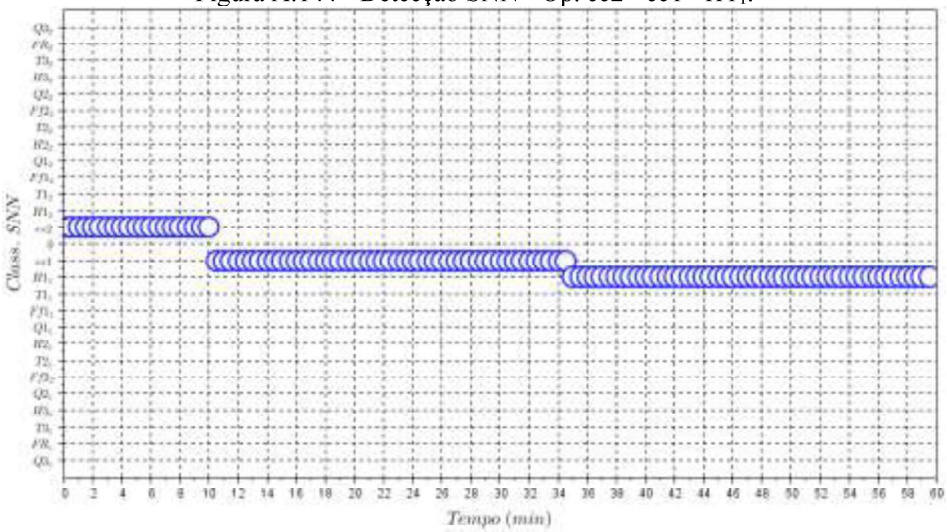


Figura A.145 - Detecção SNN - Op. *ee*1 - *ee*2 - $H1_2$.

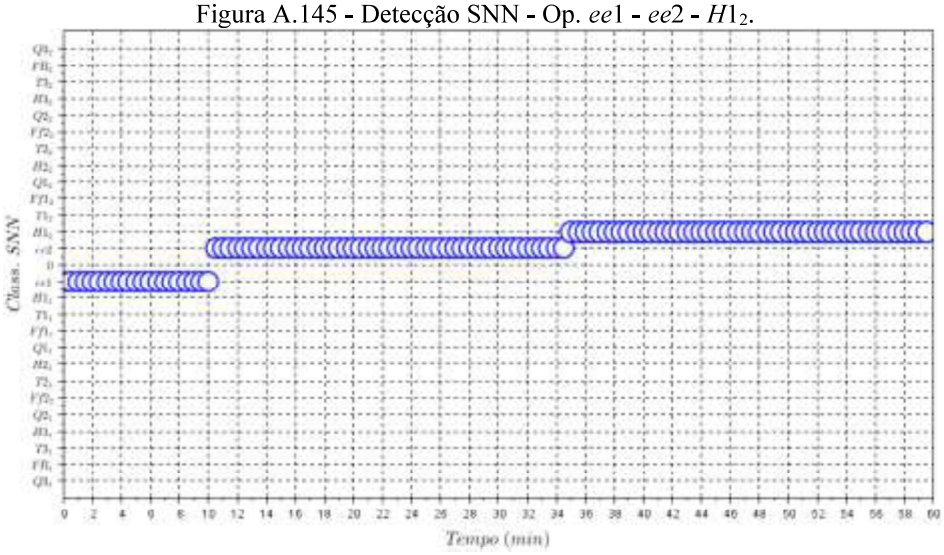


Figura A.146 - Detecção SNN - Op. *ee*2 - *ee*1 - $T1_1$.

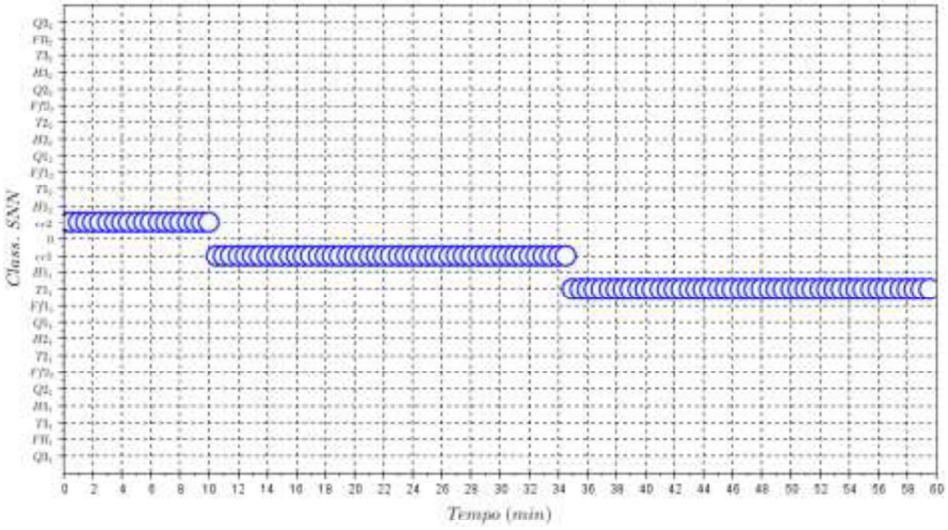


Figura A.147 - Detecção SNN - Op. *ee*1 - *ee*2 - $T1_2$.

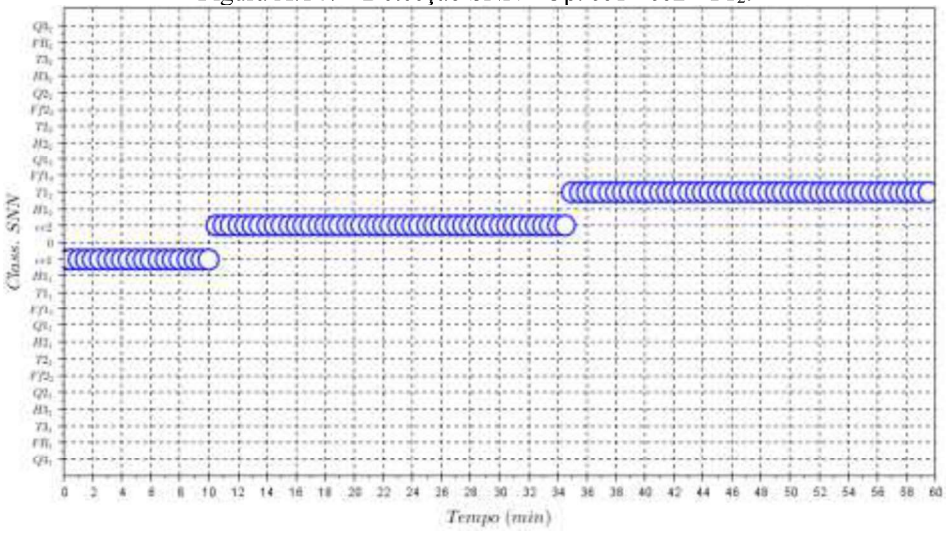


Figura A.148 - Detecção SNN - Op. *ee*2 - *ee*1 - $Ff1_1$.

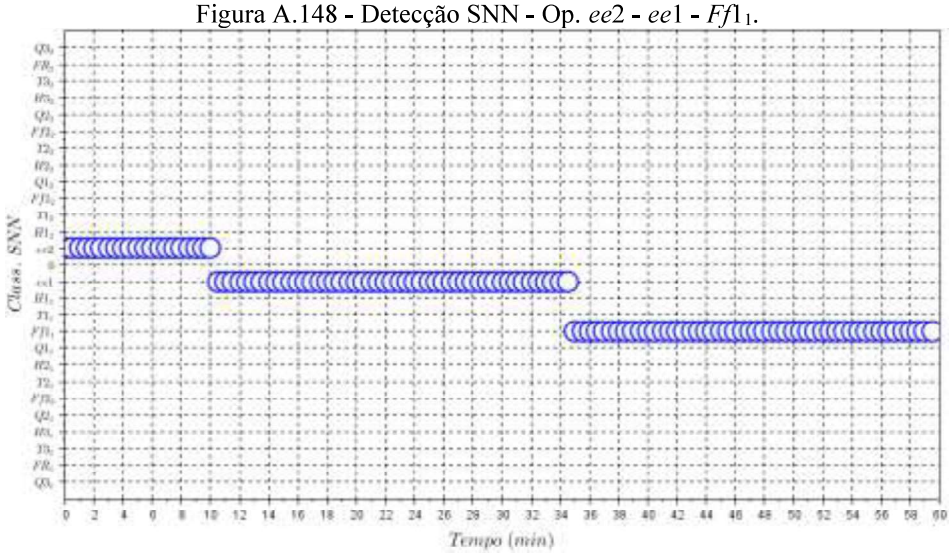


Figura A.149 - Detecção SNN - Op. *ee*1 - *ee*2 - $Ff1_2$.

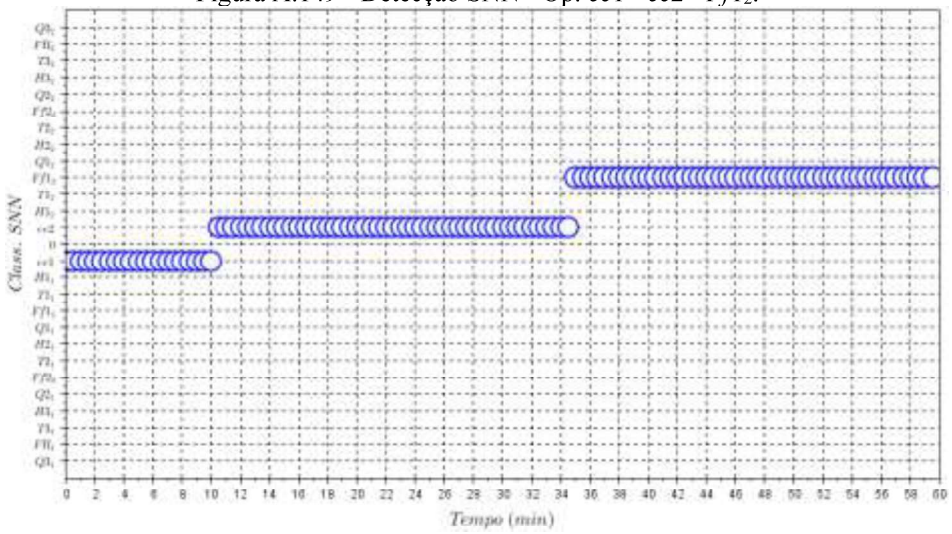


Figura A.150 - Detecção SNN - Op. *ee*2 - *ee*1 - $Q1_1$.

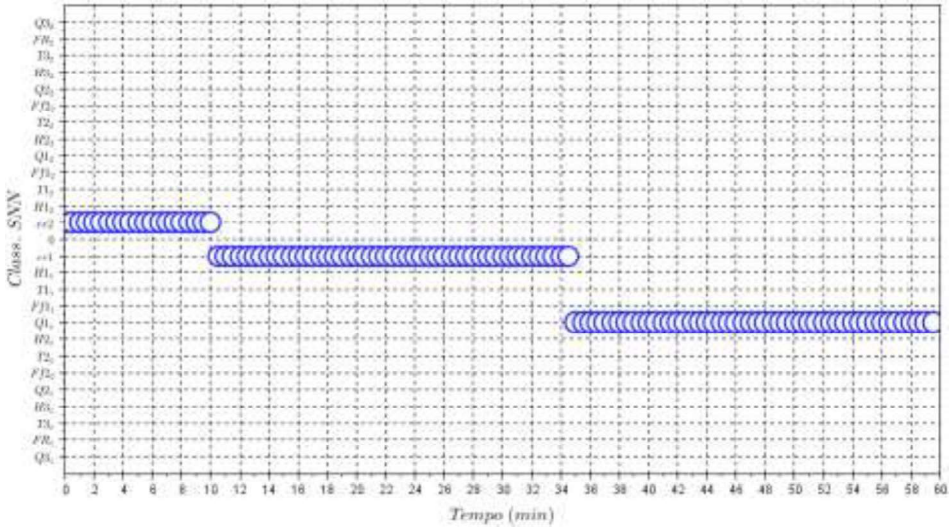


Figura A.151 - Detecção SNN - Op. *ee*1 - *ee*2 - $Q1_2$.

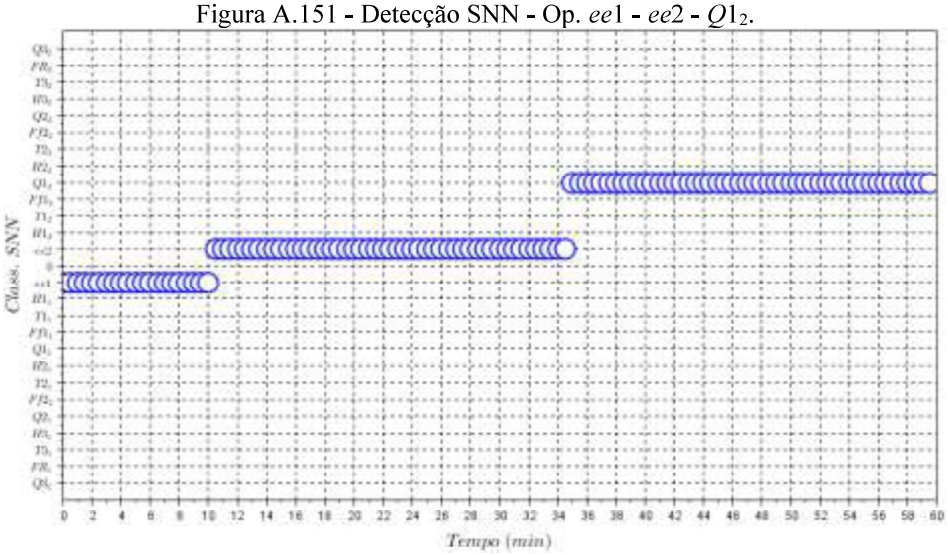


Figura A.152 - Detecção SNN - Op. *ee*2 - *ee*1 - $H2_1$.

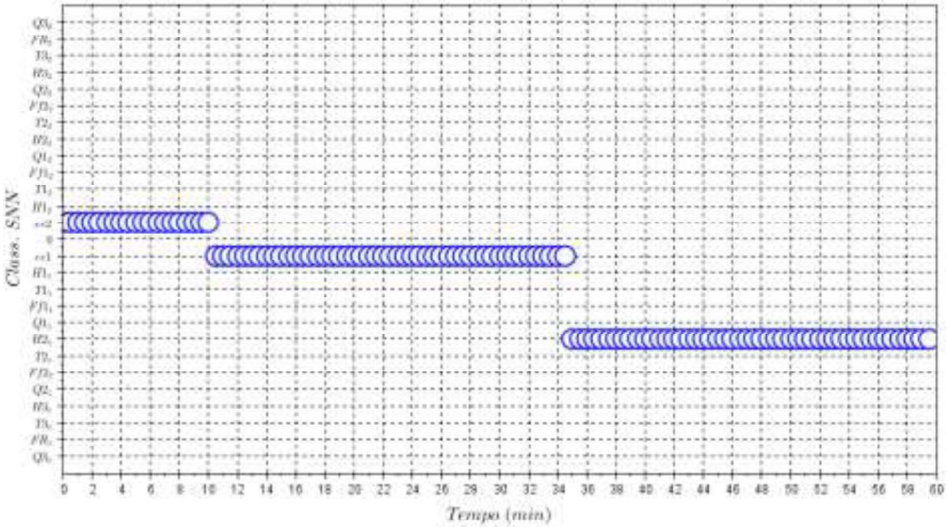


Figura A.153 - Detecção SNN - Op. *ee*1 - *ee*2 - $H2_2$.

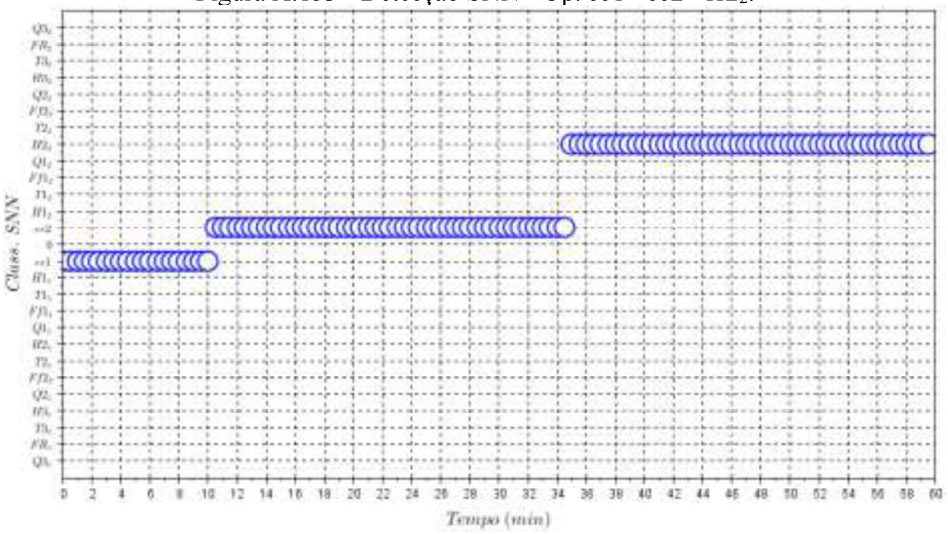


Figura A.154 - Detecção SNN - Op. *ee*2 - *ee*1 - $T2_1$.

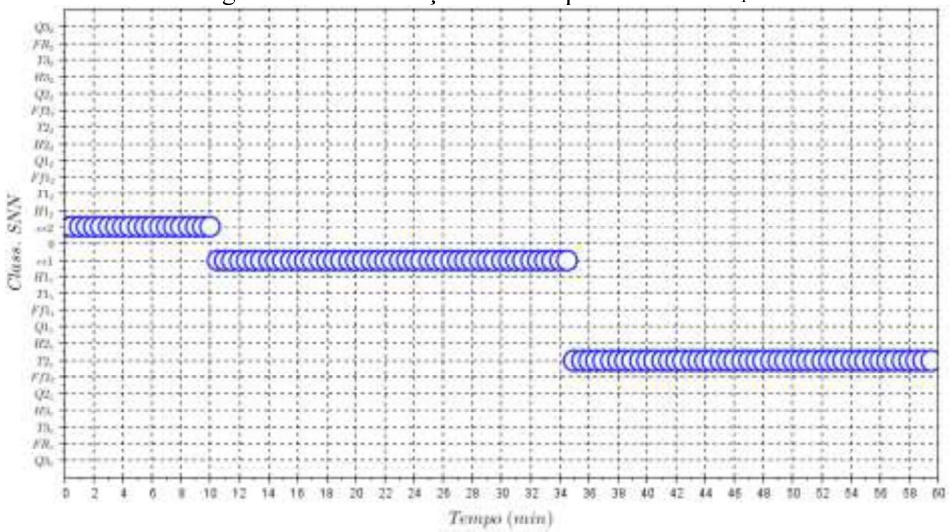


Figura A.155 - Detecção SNN - Op. *ee*1 - *ee*2 - $T2_2$.

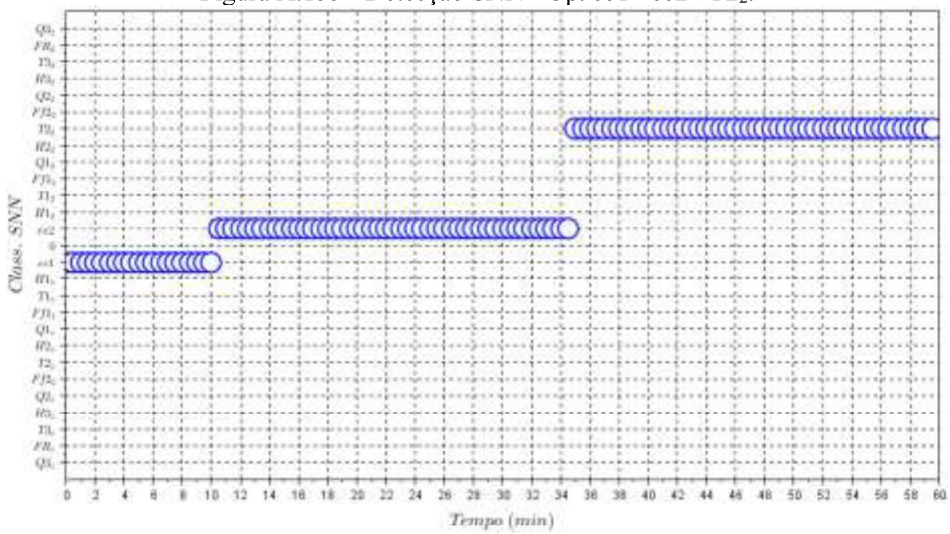


Figura A.156 - Detecção SNN - Op. *ee*2 - *ee*1 - $Ff2_1$.

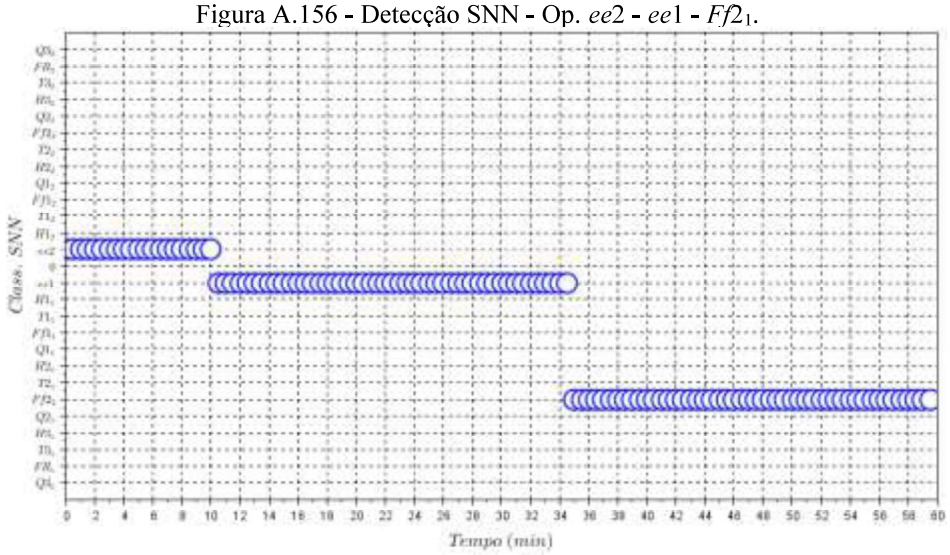


Figura A.157 - Detecção SNN - Op. *ee*1 - *ee*2 - $Ff2_2$.

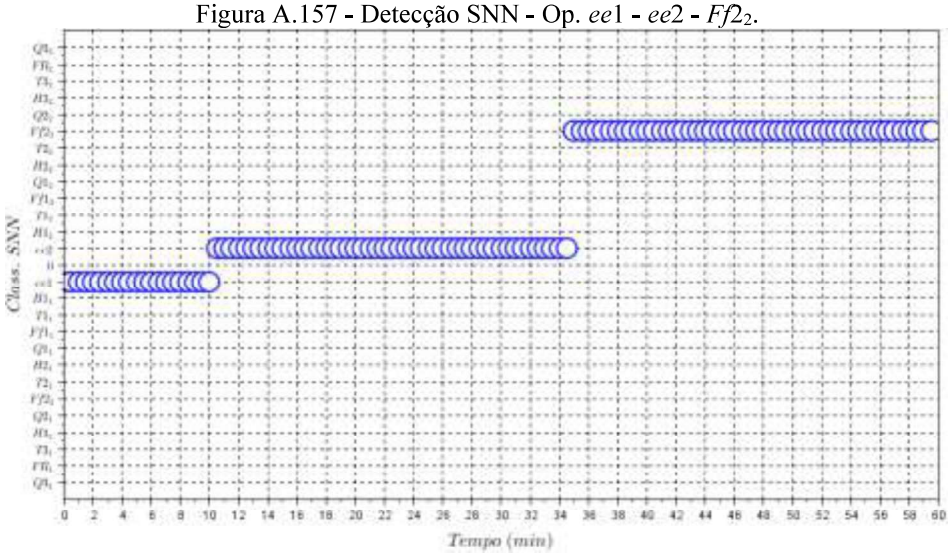


Figura A.158 - Detecção SNN - Op. *ee*2 - *ee*1 - $Q2_1$.

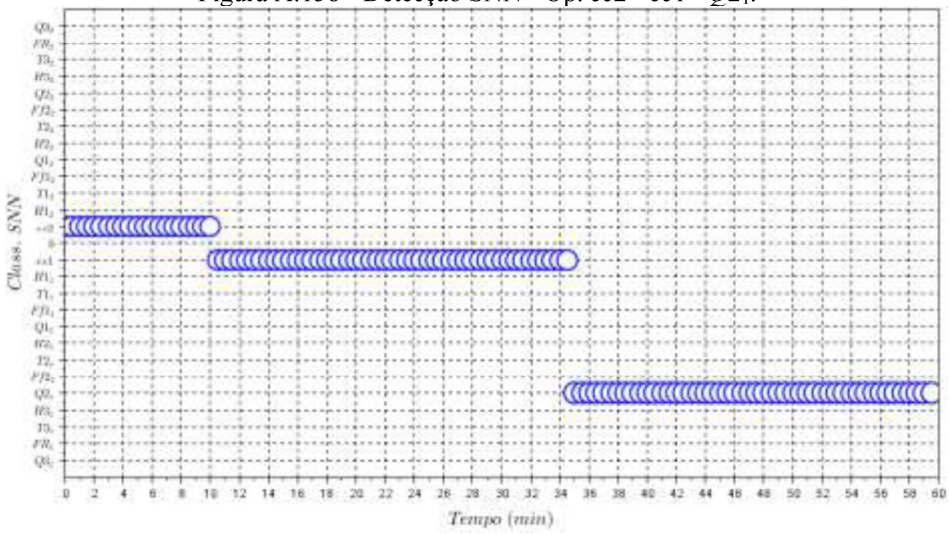


Figura A.159 - Detecção SNN - Op. *ee*1 - *ee*2 - $Q2_2$.

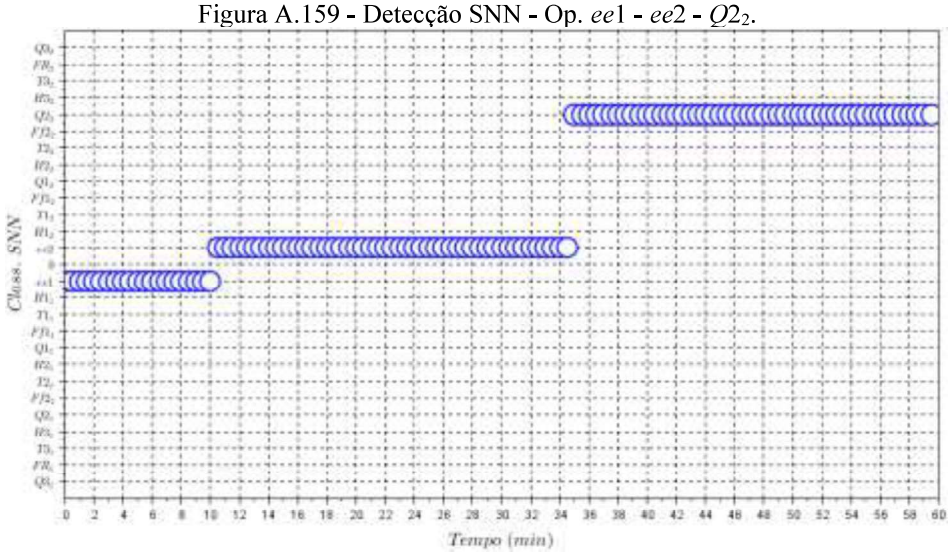


Figura A.160 - Detecção SNN - Op. *ee*2 - *ee*1 - $H3_1$.

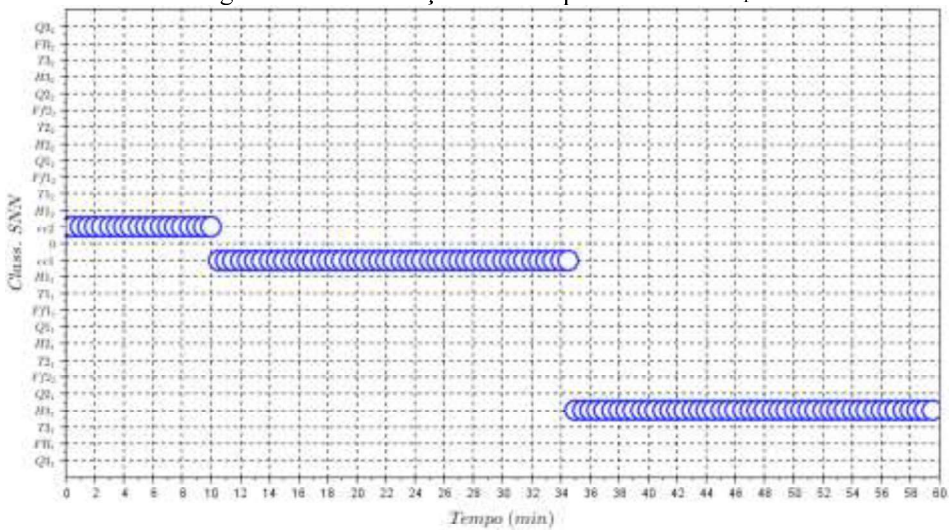


Figura A.161 - Detecção SNN - Op. *ee*1 - *ee*2 - $H3_2$.

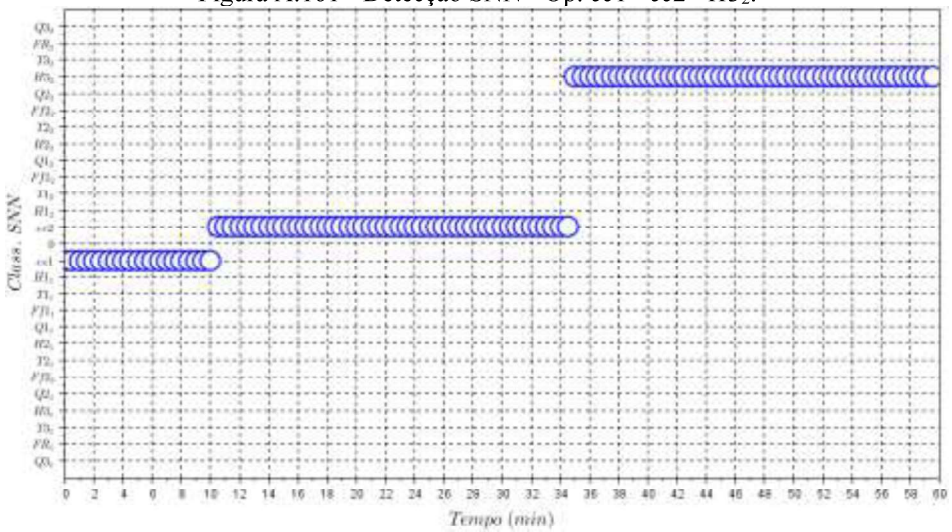


Figura A.162 - Detecção SNN - Op. *ee*2 - *ee*1 - $T3_1$.

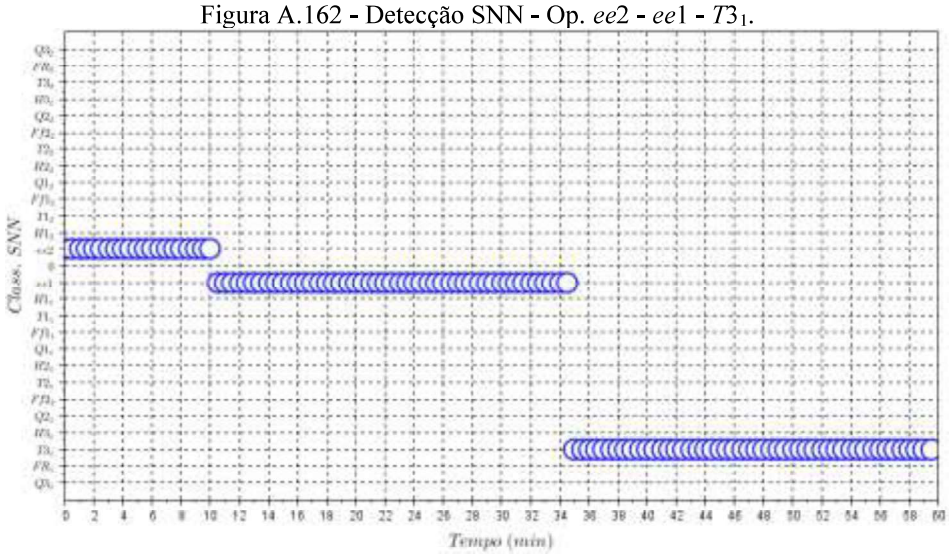


Figura A.163 - Detecção SNN - Op. *ee*1 - *ee*2 - $T3_2$.

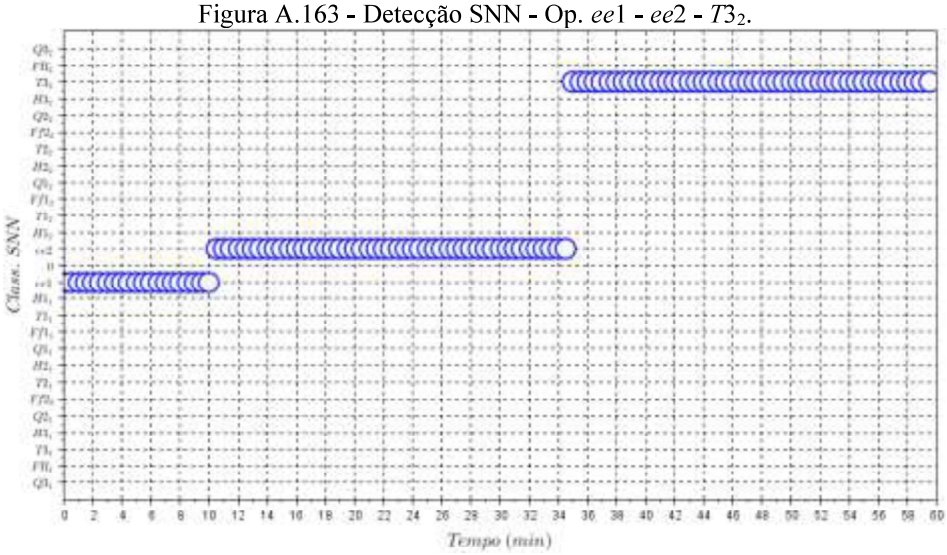


Figura A.164 - Detecção SNN - Op. *ee*2 - *ee*1 - F_{R1}.

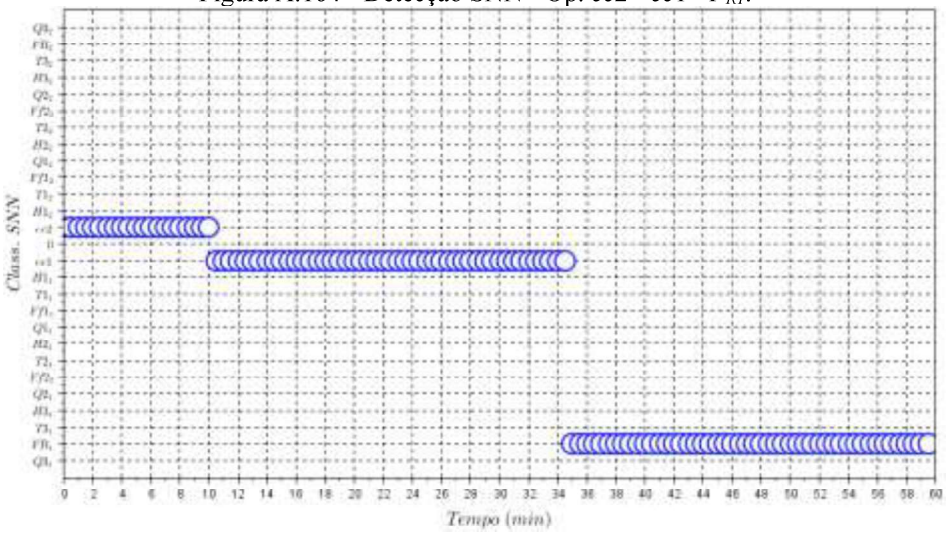


Figura A.165 - Detecção SNN - Op. *ee*1 - *ee*2 - F_{R2}.

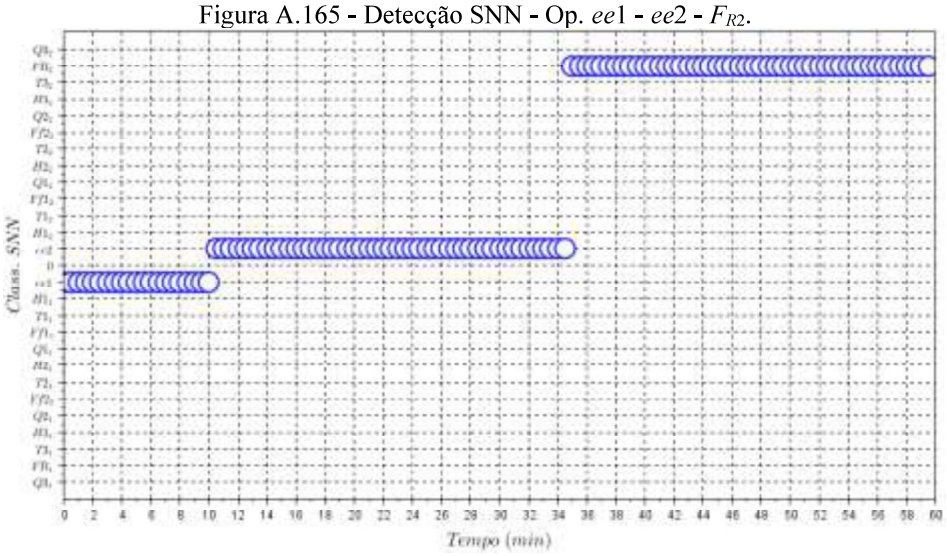


Figura A.166 - Detecção SNN - Op. *ee*2 - *ee*1 - $Q3_1$.

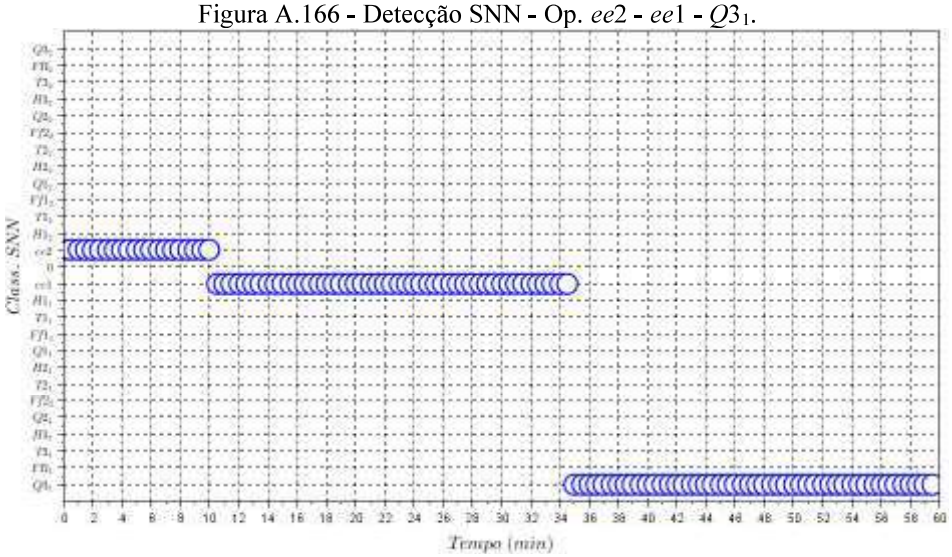
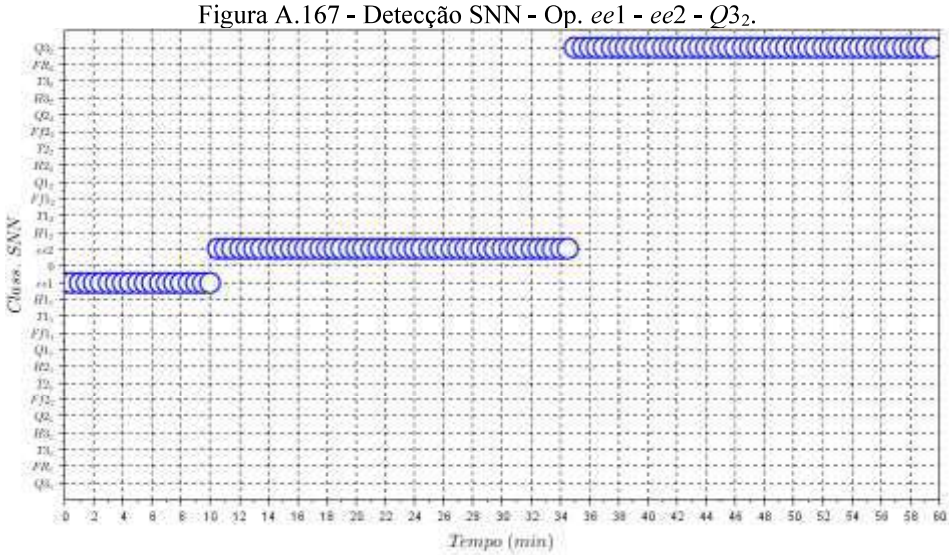


Figura A.167 - Detecção SNN - Op. *ee*1 - *ee*2 - $Q3_2$.



A.7. Resultados DNN

Foram estabelecidas 24 falhas, sendo duas falhas para cada variável, aplicadas em cada um dos dois estados estacionários estipulados, de acordo com o Capítulo 3. Além das 24 falhas, foram utilizados dados de ambos os estados estacionários, totalizando 26 situações passíveis de detecção. A metodologia de detecção de falhas DNN utiliza os valores de entrada (variáveis manipuladas e variáveis controladas) de forma que correspondam as probabilidades de cada conjunto de dados fazer ou não parte de uma determinada classe, assim como as redes neuronais. A diferença se encontra no número de camadas ocultas da rede, que no caso da SNN é somente uma única camada. Para a rede DNN foram estabelecidas cinco camadas com o dobro de nós da camada de saída em cada camada, ou seja, 52 nós por camada. Foram estabelecidas 1000 (mil) épocas para o treinamento da rede DNN.

Para todas as situações passíveis de detecção, foram simulados computacionalmente 26 conjuntos de sinais dispostos em uma matriz, em que outro estado estacionário ou falha foram aplicados nos instantes 10 minutos e 35 minutos, em um total de 60 minutos de operação, para cada sinal treinado. Os dados então foram separados em dados de treinamento e de validação. Foi utilizada uma proporção de 2 para 1 entre os dados de treinamento e validação. A cada dois instantes separados para treinamento, o terceiro instante era designado para validação. Os resultados da detecção para cada uma das variáveis e suas falhas são apresentados nas figuras a seguir.

Figura A.168 - Detecção DNN - Op. *ee*2 - *ee*1 - *ee*2.

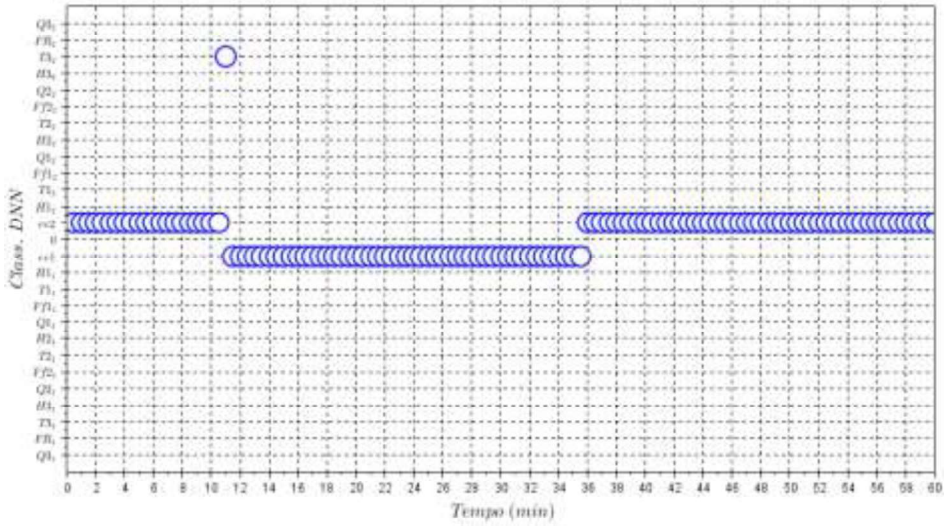


Figura A.169 - Detecção DNN - Op. *ee*1 - *ee*2 - *ee*1.

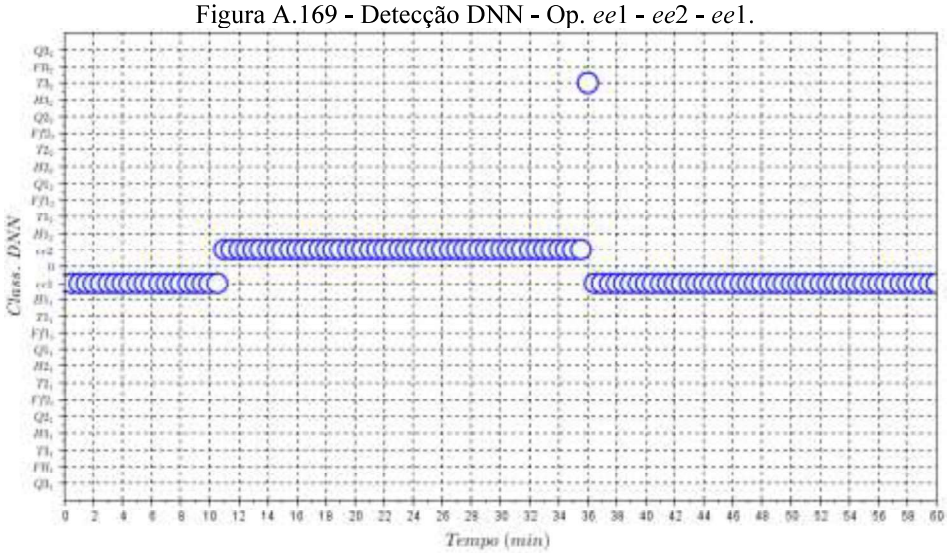


Figura A.170 - Detecção DNN - Op. *ee*2 - *ee*1 - $H1_1$.

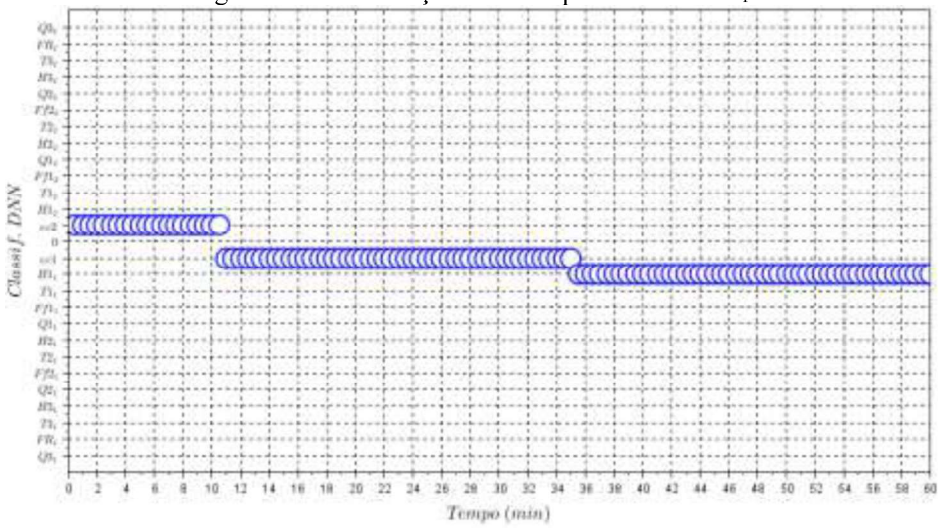


Figura A.171 - Detecção DNN - Op. *ee*1 - *ee*2 - $H1_2$.

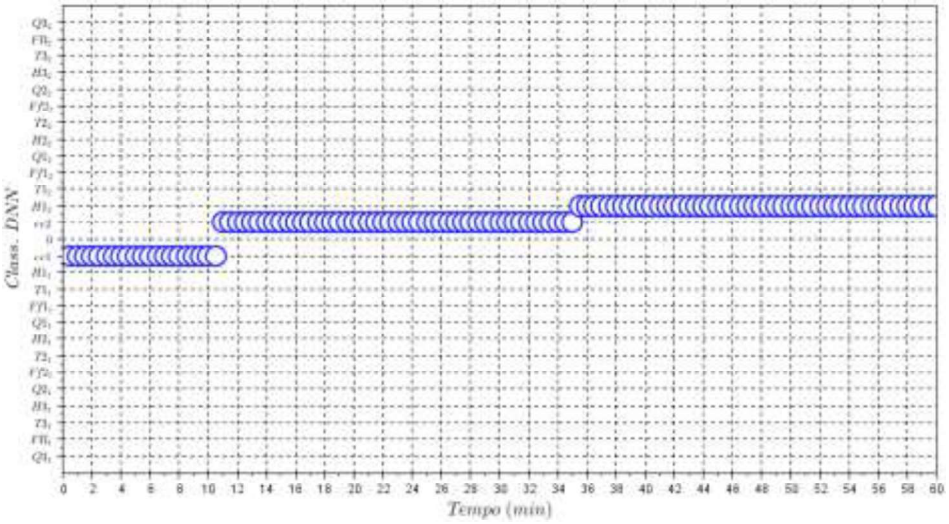


Figura A.172 - Detecção DNN - Op. *ee*2 - *ee*1 - $T1_1$.

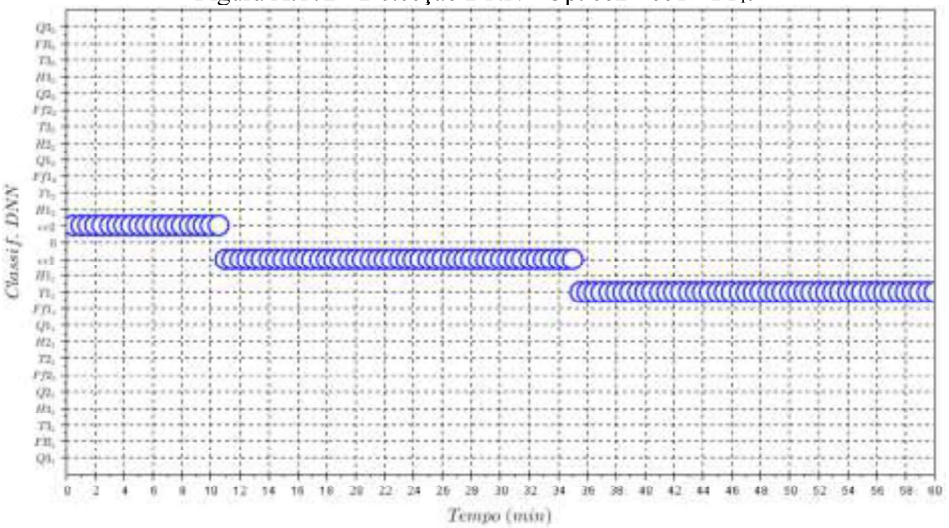


Figura A.173 - Detecção DNN - Op. *ee*1 - *ee*2 - Tl_2.

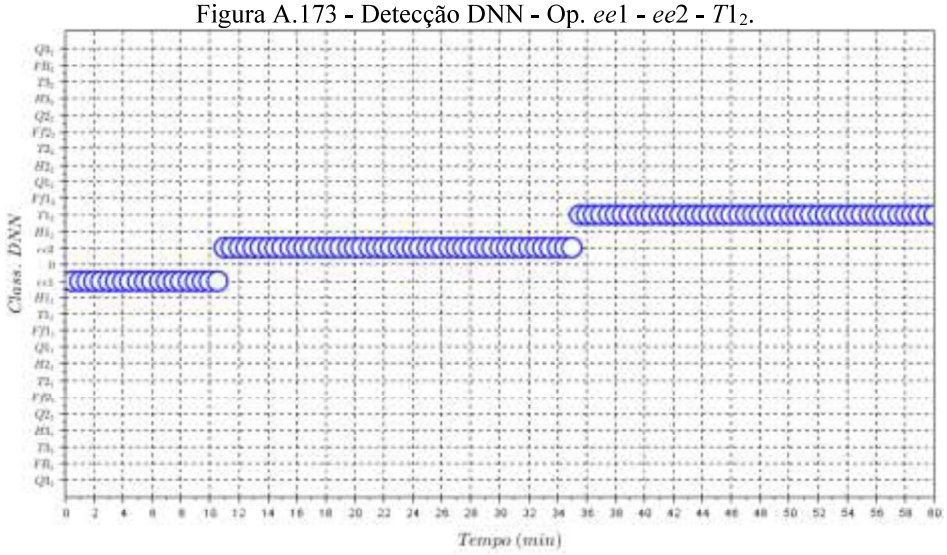


Figura A.174 - Detecção DNN - Op. *ee*2 - *ee*1 - $Ff1_1$.

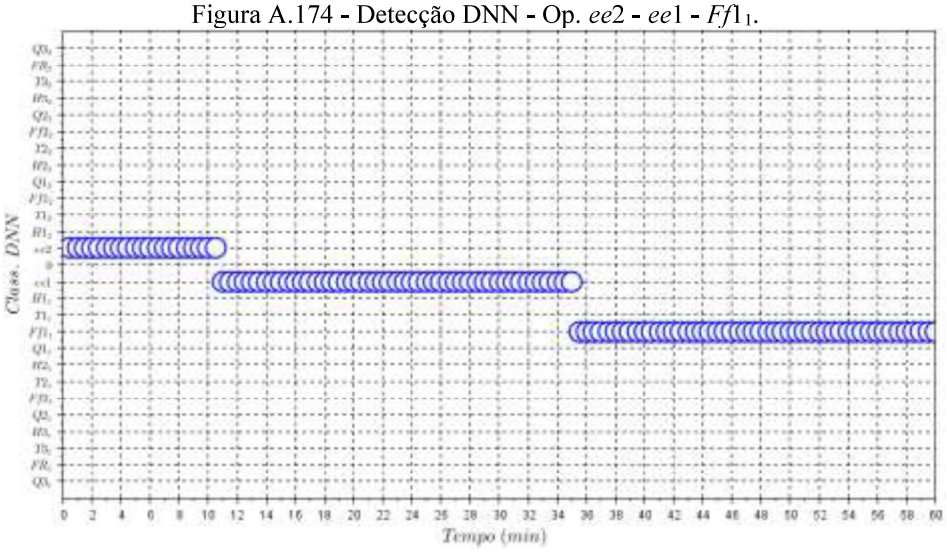


Figura A.175 - Detecção DNN - Op. *ee*1 - *ee*2 - $Ff1_2$.

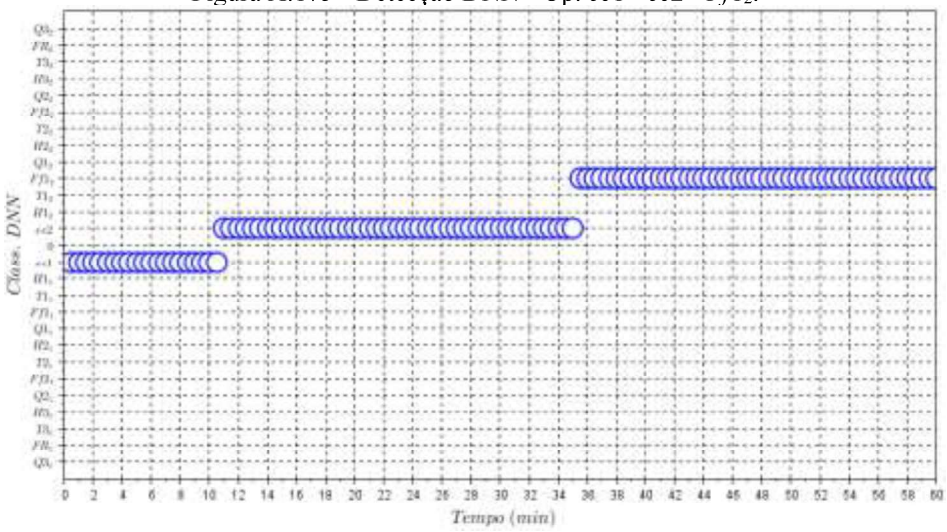


Figura A.176 - Detecção DNN - Op. *ee*2 - *ee*1 - $Q1_1$.

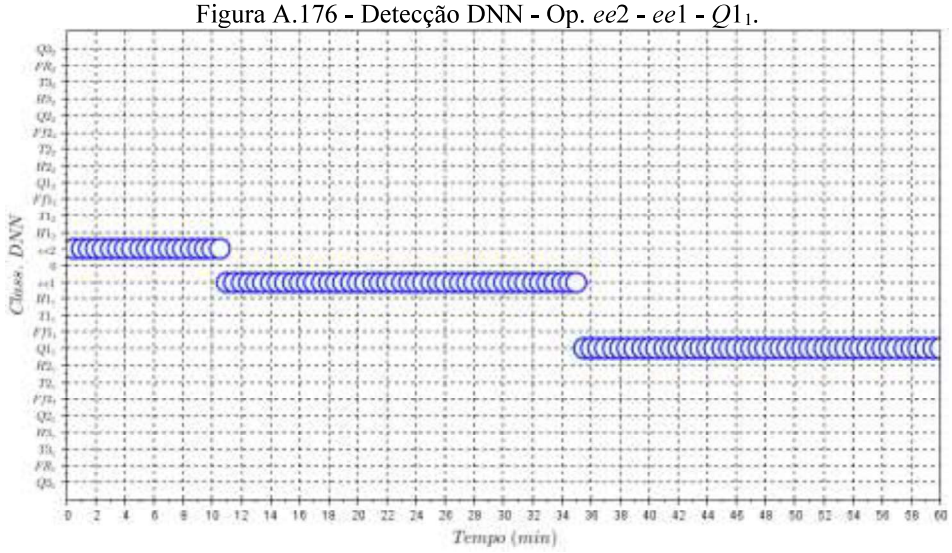


Figura A.177 - Detecção DNN - Op. *ee*1 - *ee*2 - $Q1_2$.

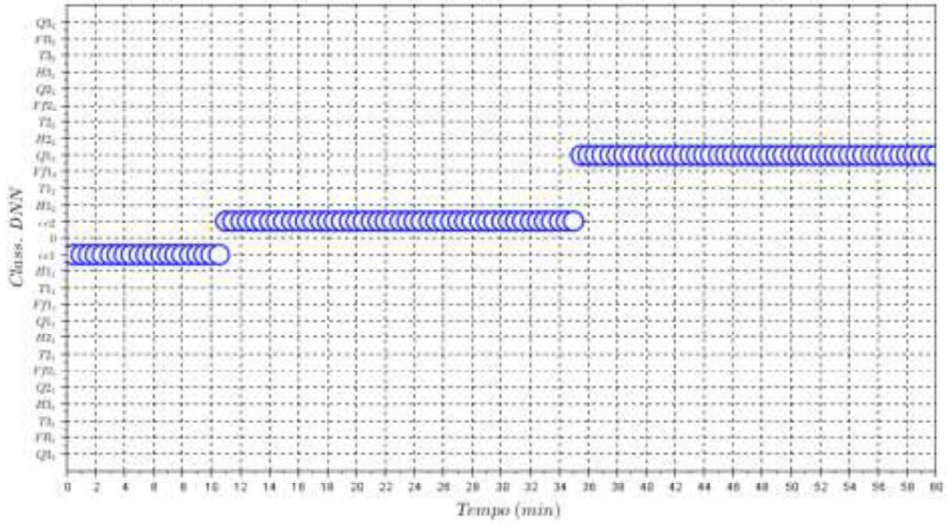


Figura A.178 - Detecção DNN - Op. *ee*2 - *ee*1 - $H2_1$.

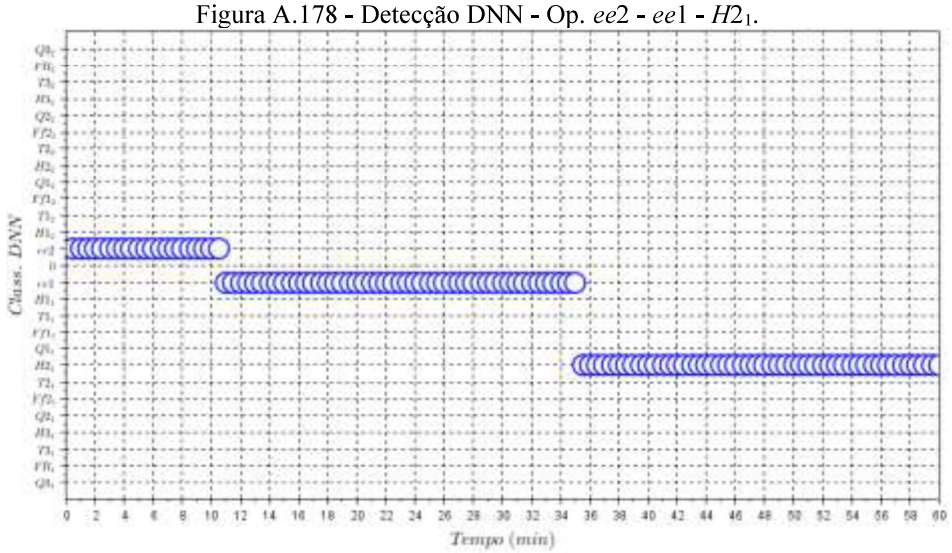


Figura A.179 - Detecção DNN - Op. *ee*1 - *ee*2 - $H2_2$.

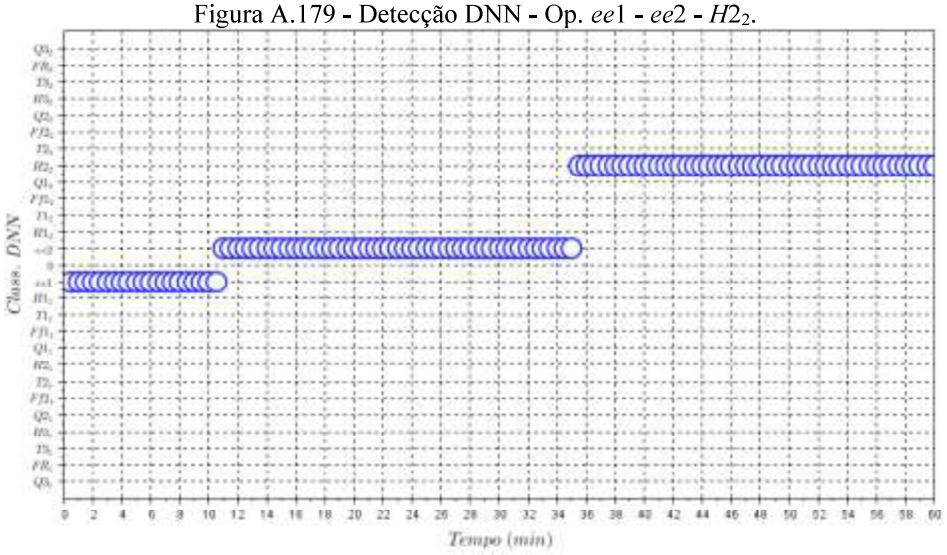


Figura A.180 - Detecção DNN - Op. *ee*2 - *ee*1 - $T2_1$.

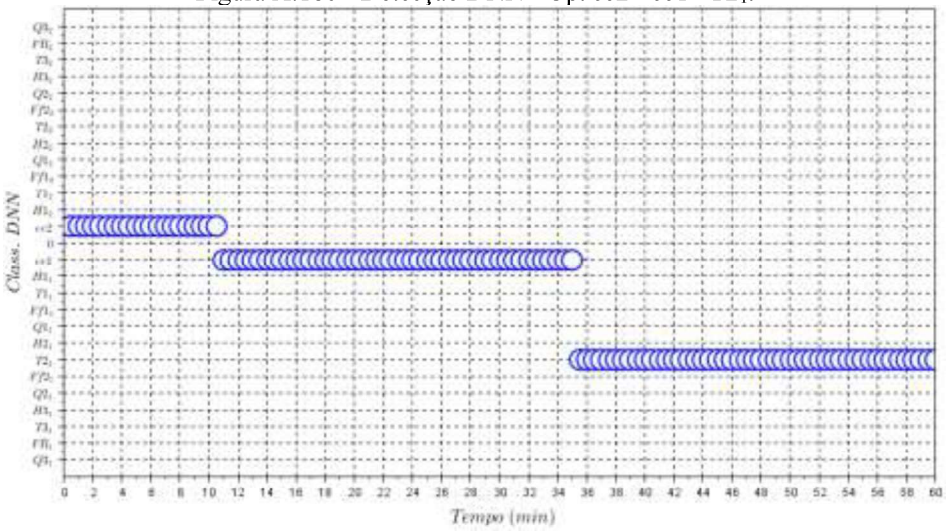


Figura A.181 - Detecção DNN - Op. *ee*1 - *ee*2 - $T2_2$.

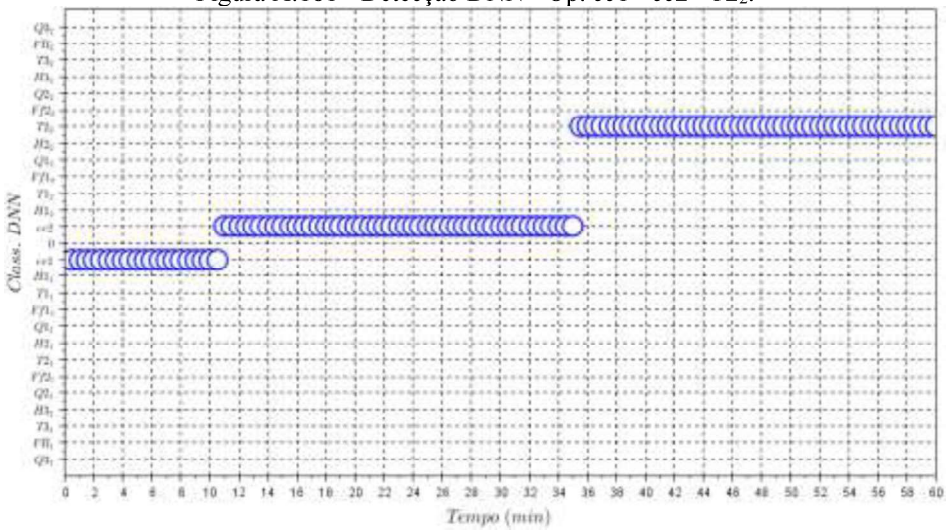


Figura A.182 - Detecção DNN - Op. *ee*2 - *ee*1 - $Ff2_1$.

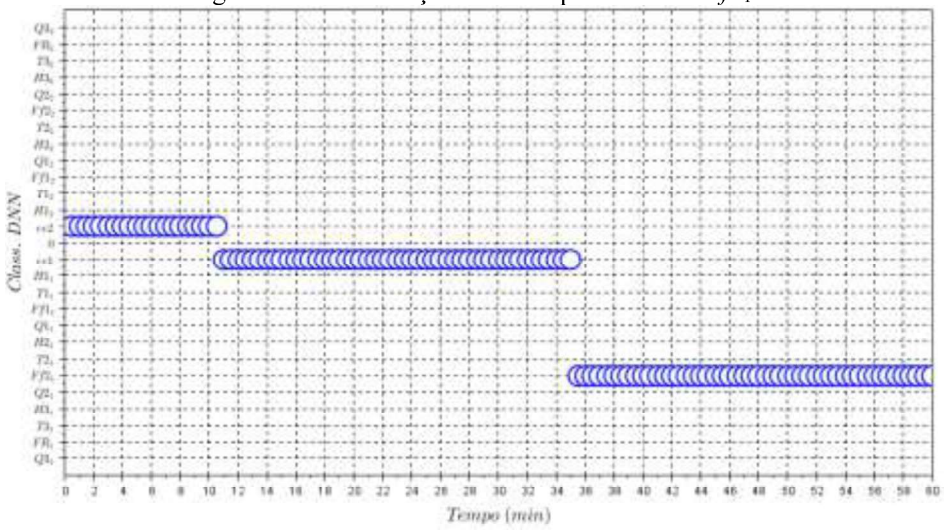


Figura A.183 - Detecção DNN - Op. *ee*1 - *ee*2 - $Ff2_2$.

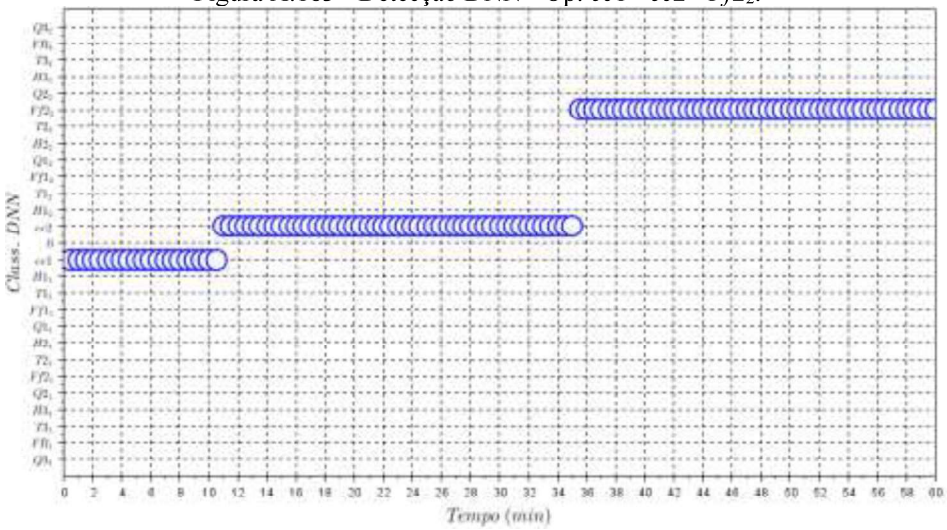


Figura A.184 - Detecção DNN - Op. *ee*2 - *ee*1 - $Q2_1$.

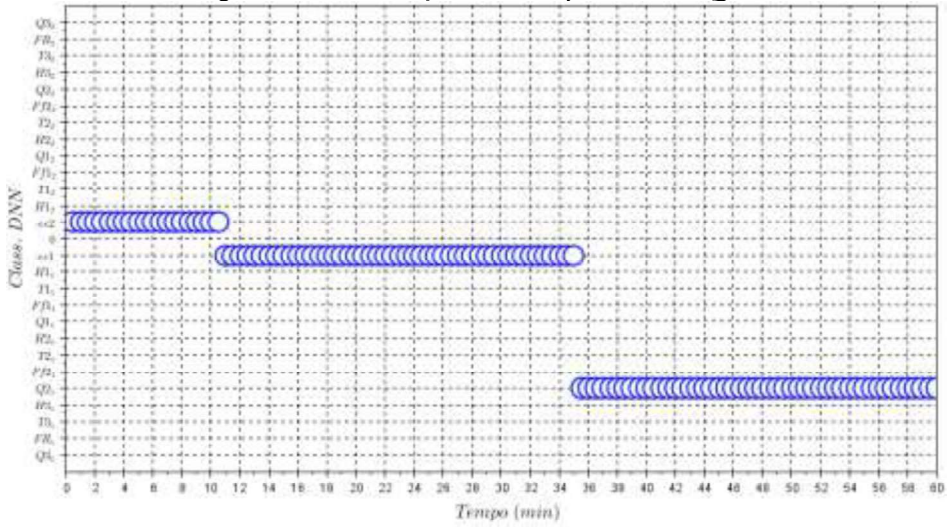


Figura A.185 - Detecção DNN - Op. *ee*1 - *ee*2 - $Q2_2$.

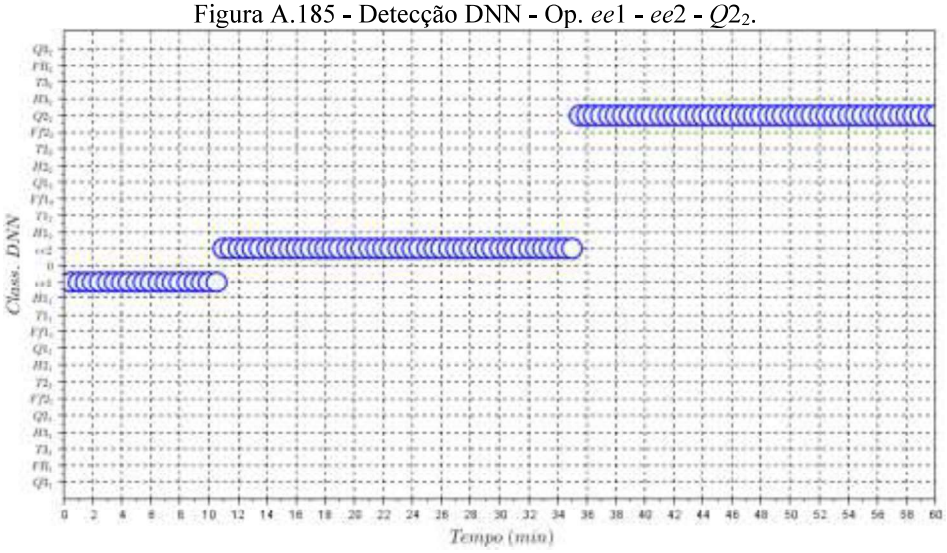


Figura A.186 - Detecção DNN - Op. *ee*2 - *ee*1 - $H3_1$.

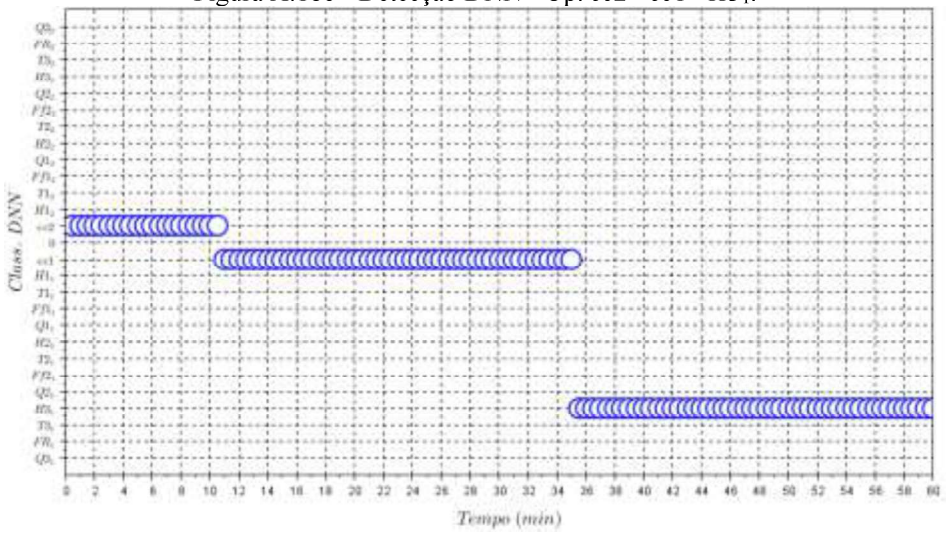


Figura A.187 - Detecção DNN - Op. *ee*1 - *ee*2 - $H3_2$.

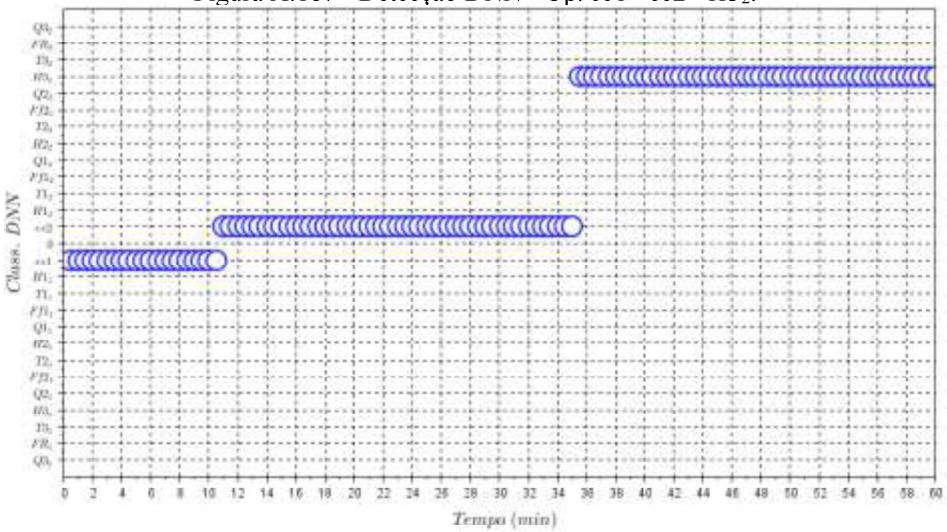


Figura A.188 - Detecção DNN - Op. *ee*2 - *ee*1 - $T3_1$.

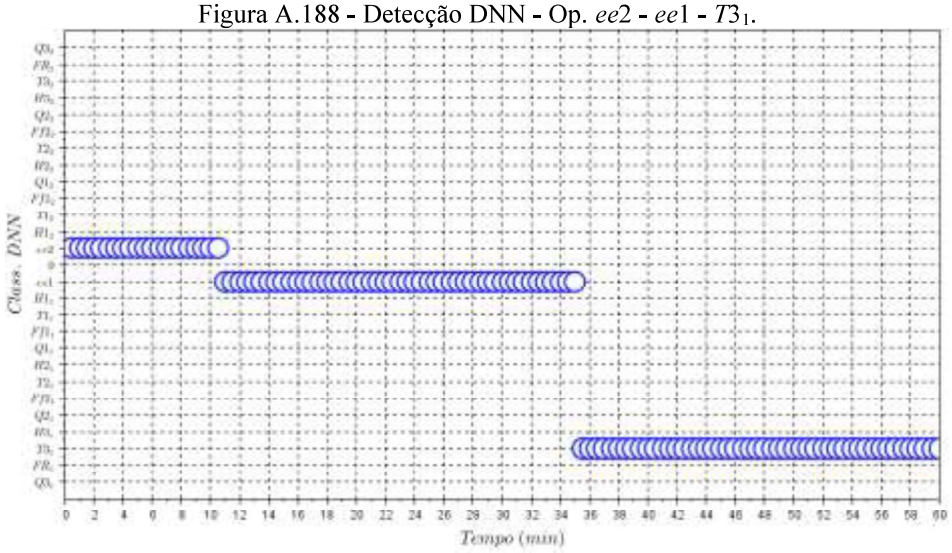


Figura A.189 - Detecção DNN - Op. *ee*1 - *ee*2 - $T3_2$.

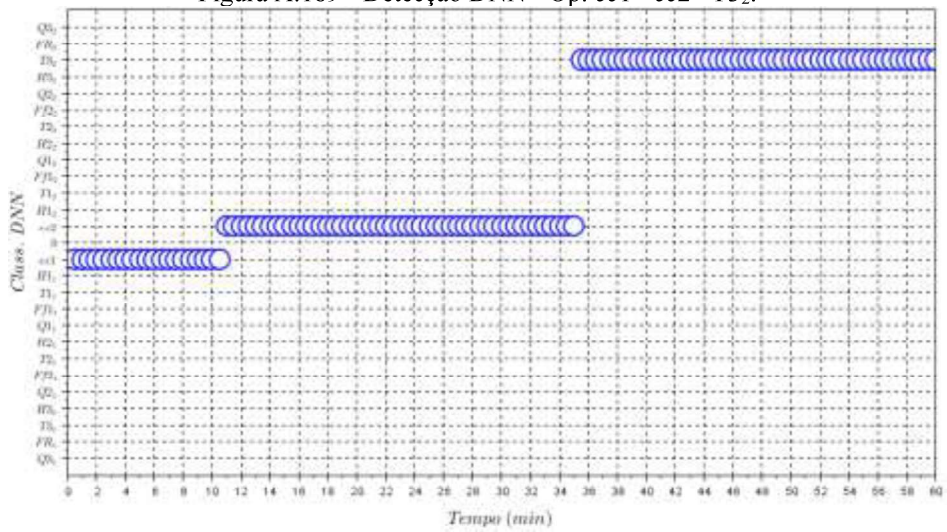


Figura A.190 - Detecção DNN - Op. *ee*2 - *ee*1 - F_{R1}.

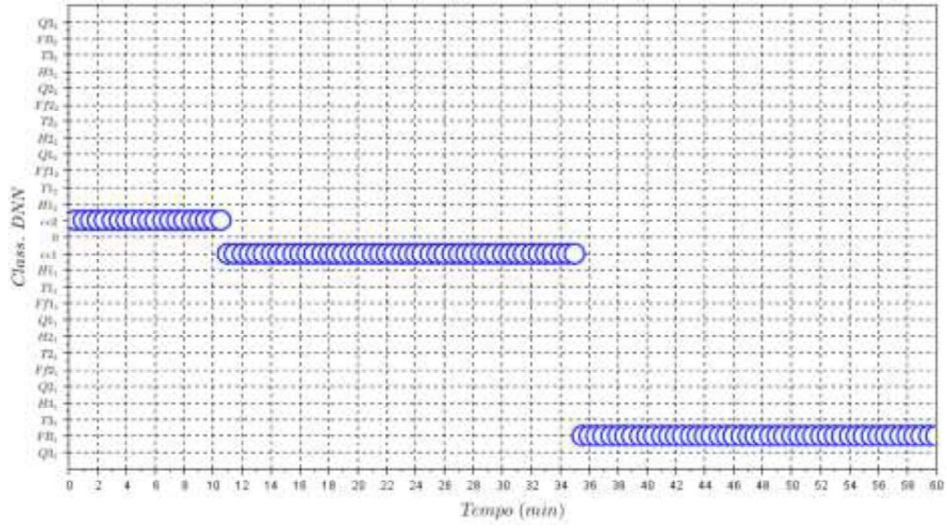


Figura A.191 - Detecção DNN - Op. *ee*1 - *ee*2 - F_{R2}.

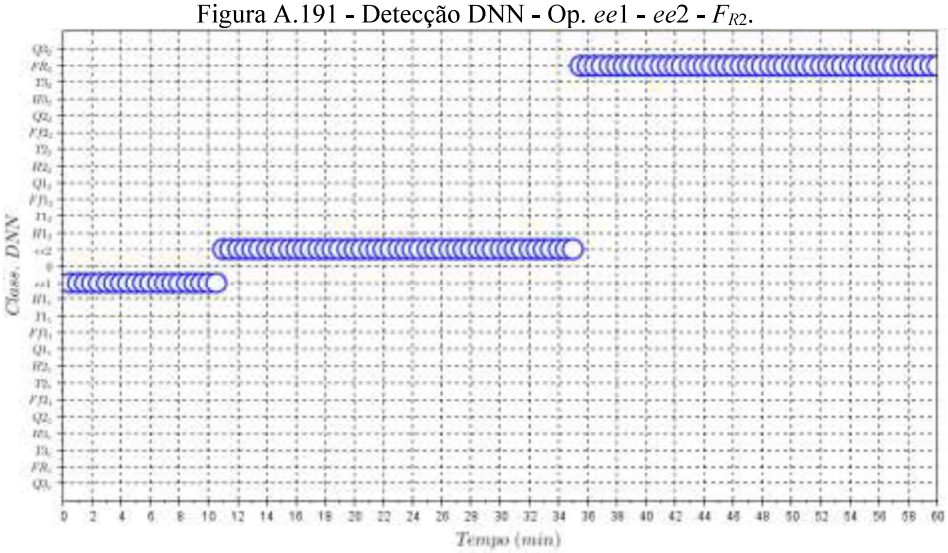


Figura A.192 - Detecção DNN - Op. *ee*2 - *ee*1 - $Q3_1$.

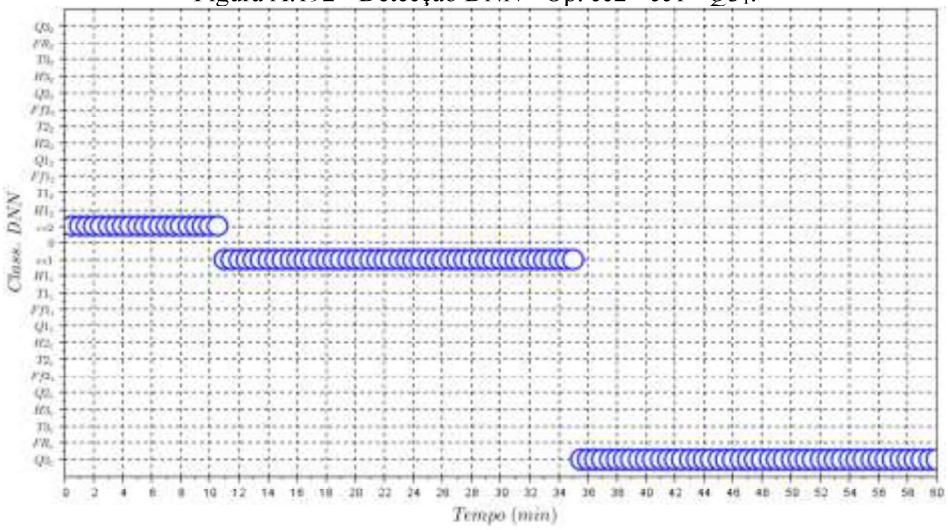
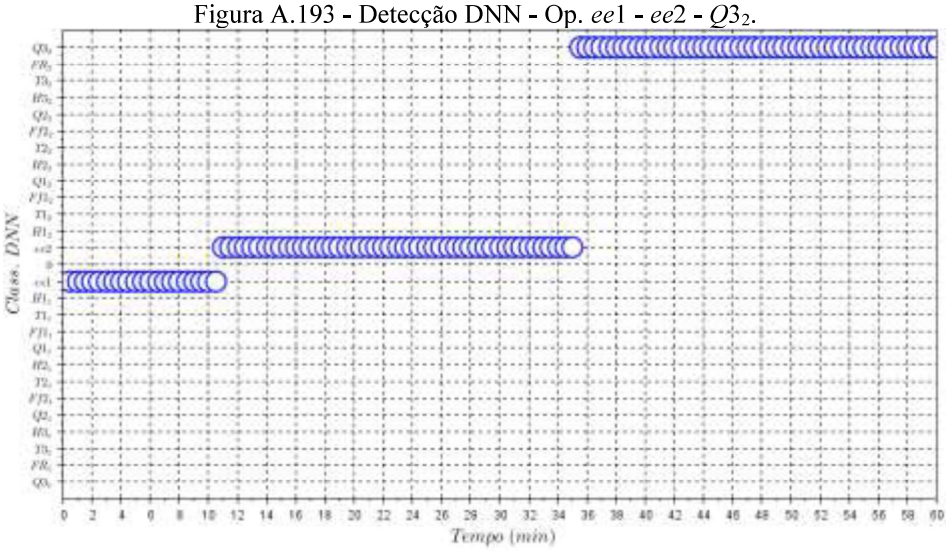


Figura A.193 - Detecção DNN - Op. *ee*1 - *ee*2 - $Q3_2$.



A.8. Resultados LNN

Foram estabelecidas 24 falhas, sendo duas falhas para cada variável, aplicadas em cada um dos dois estados estacionários estipulados, de acordo com o Capítulo 3. Além das 24 falhas, foram utilizados dados de ambos os estados estacionários, totalizando 26 situações passíveis de detecção. A metodologia de detecção de falhas LNN utiliza cada valor de entrada (variáveis manipuladas e variáveis controladas) de forma que sejam classificadas de acordo com variáveis linguísticas: Muito Alto (MA), Alto (A), Baixo (B) e Muito Baixo (MB), através de lógica *Fuzzy C-Means*. A saída dos sistemas *Fuzzy* são as entradas da rede neuronal com uma única camada oculta, composta com o dobro de nós da saída, sendo que a saída é composta pelo número de operações a serem classificadas. Foram estabelecidas 1000 (mil) épocas para o treinamento da rede LNN.

Para todas as situações passíveis de detecção, foram simulados computacionalmente 26 conjuntos de sinais dispostos em uma matriz, em que outro estado estacionário ou falha foram aplicados nos instantes 10 minutos e 35 minutos, em um total de 60 minutos de operação, para cada sinal treinado. Os dados então foram separados em dados de treinamento e de validação. Foi utilizada uma proporção de 2 para 1 entre os dados de treinamento e validação. A cada dois instantes separados para treinamento, o terceiro instante era designado para validação. Os resultados da detecção para cada uma das variáveis e suas falhas são apresentados nas figuras a seguir.

Figura A.194 - Detecção LNN - Op. *ee*2 - *ee*1 - *ee*2.

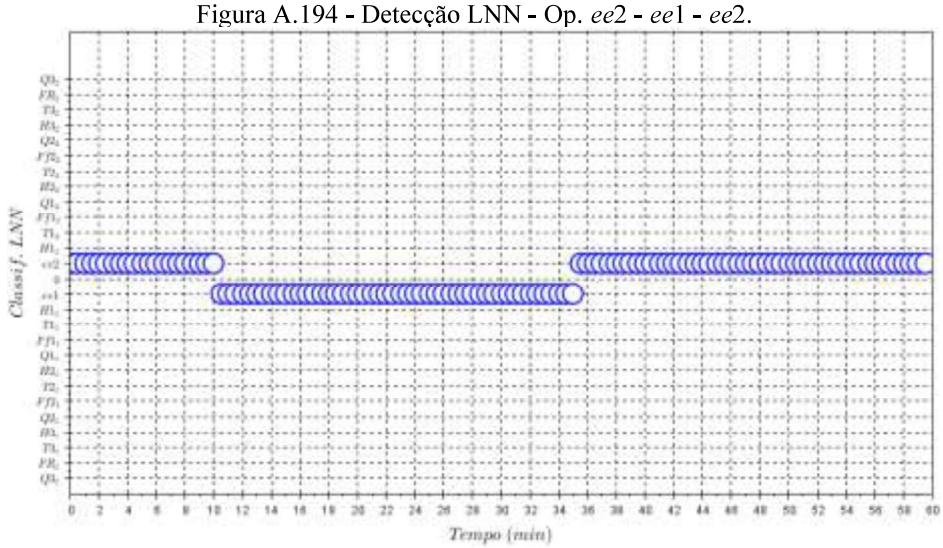


Figura A.195 - Detecção LNN - Op. *ee*1 - *ee*2 - *ee*1.

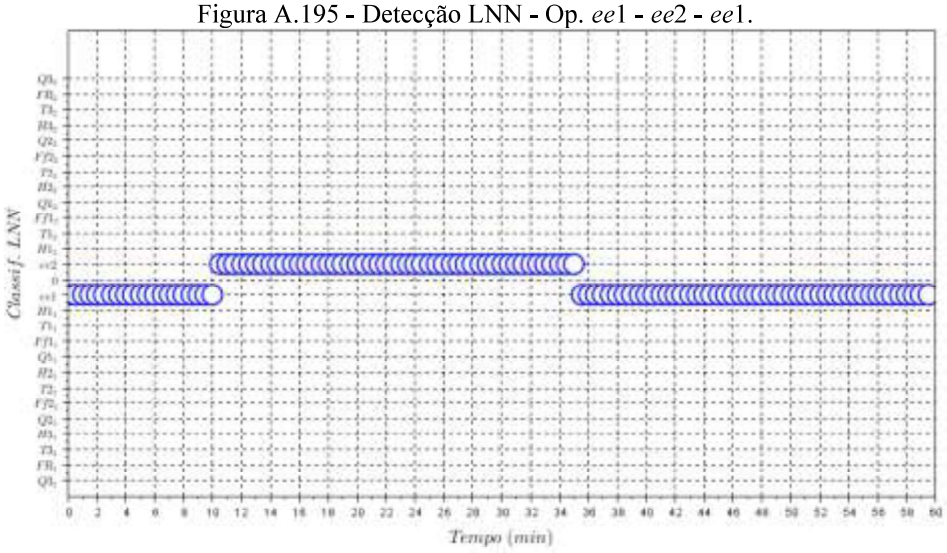


Figura A.196 - Detecção LNN - Op. *ee*2 - *ee*1 - $H1_1$.

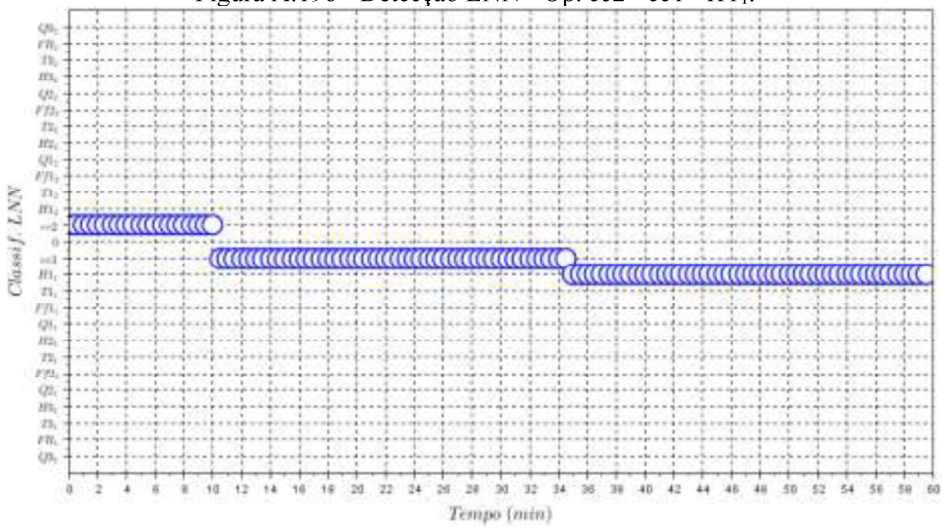


Figura A.197 - Detecção LNN - Op. *ee*1 - *ee*2 - $H1_2$.

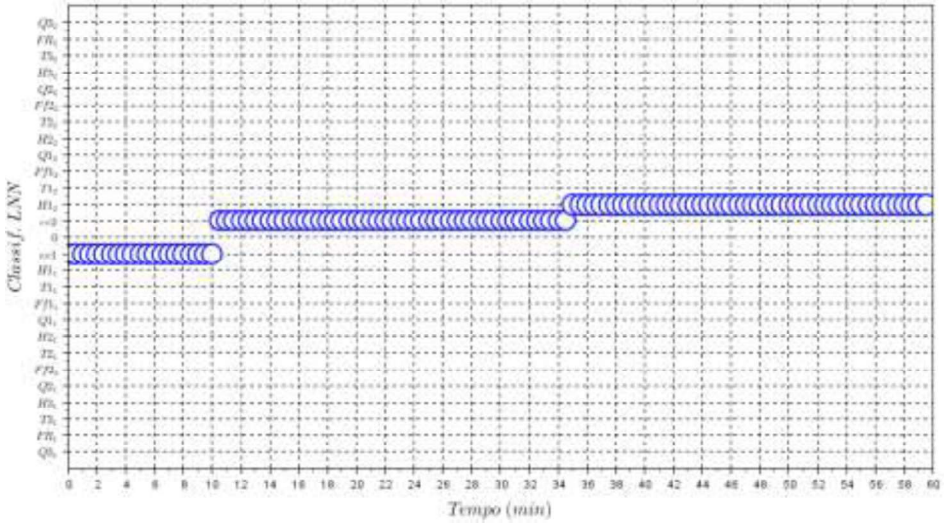


Figura A.198 - Detecção LNN - Op. *ee*2 - *ee*1 - $T1_1$.

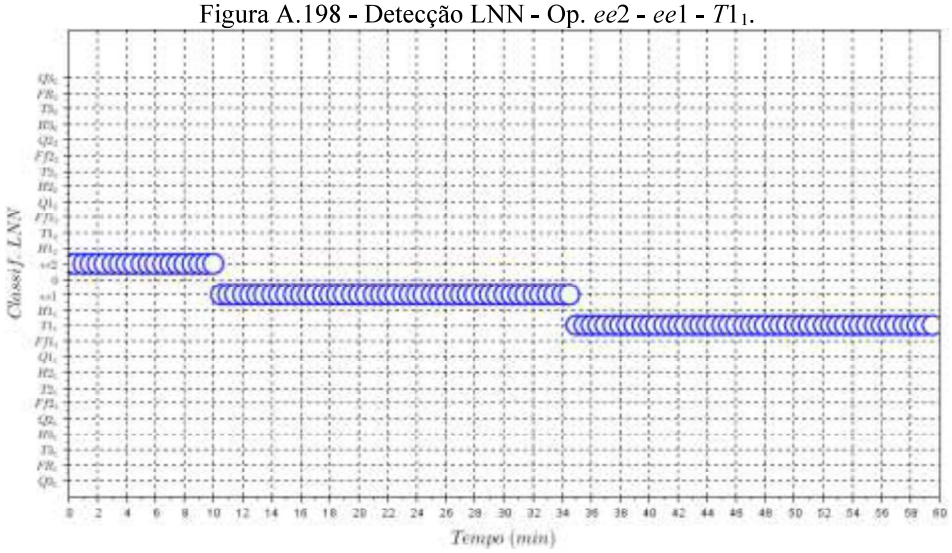


Figura A.199 - Detecção LNN - Op. *ee*1 - *ee*2 - $T1_2$.

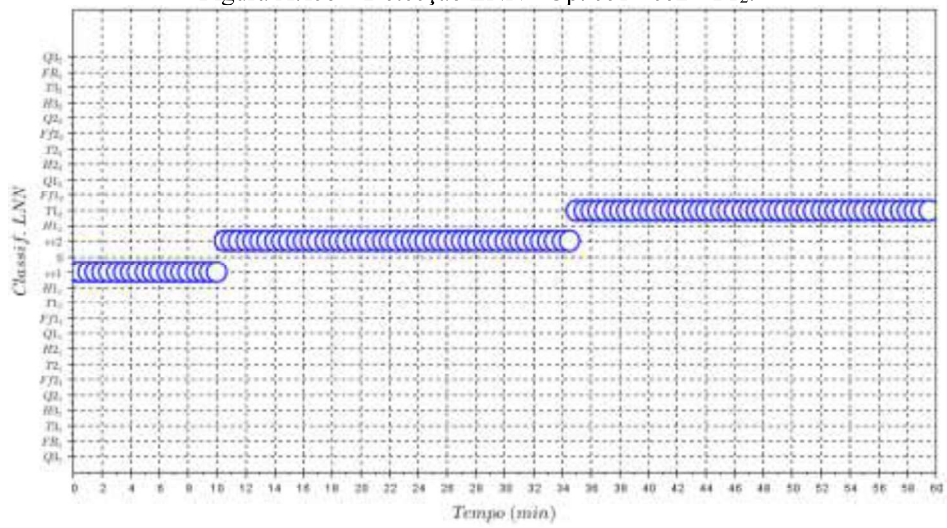


Figura A.200 - Detecção LNN - Op. *ee*2 - *ee*1 - $Ff1_1$.

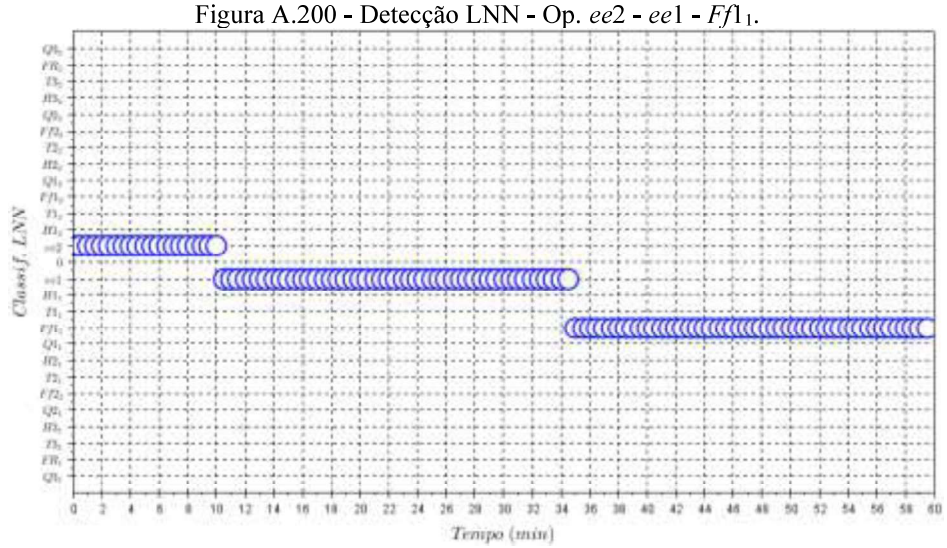


Figura A.201 - Detecção LNN - Op. *ee*1 - *ee*2 - $Ff1_2$.

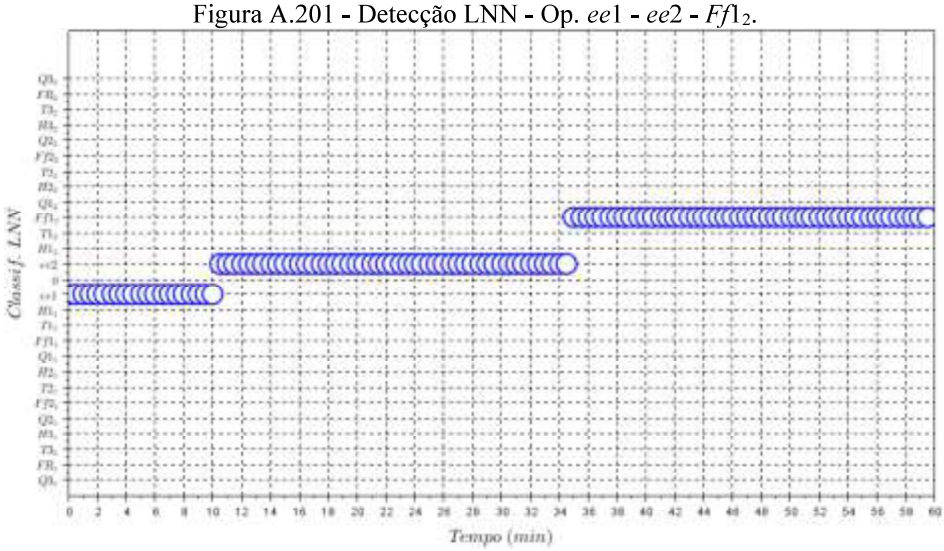


Figura A.202 - Detecção LNN - Op. *ee*2 - *ee*1 - $Q1_1$.

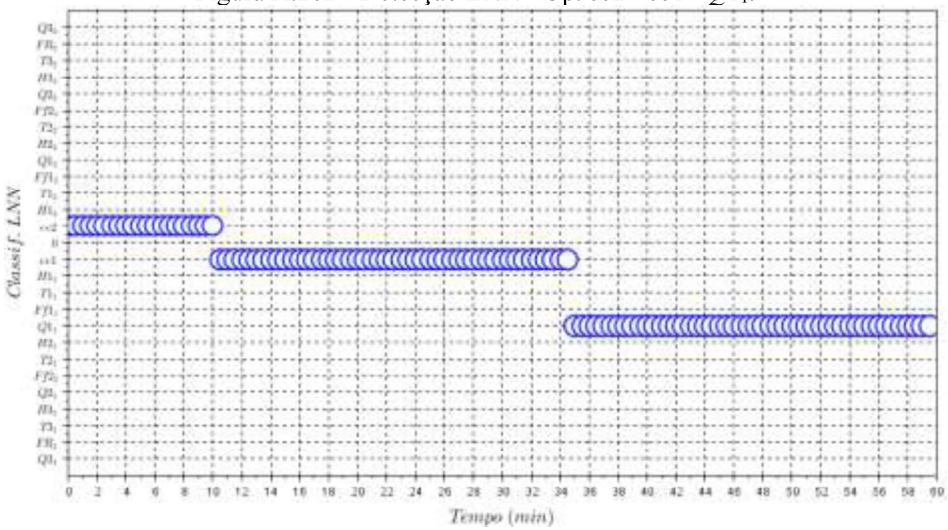


Figura A.203 - Detecção LNN - Op. *ee*1 - *ee*2 - $Q1_2$.

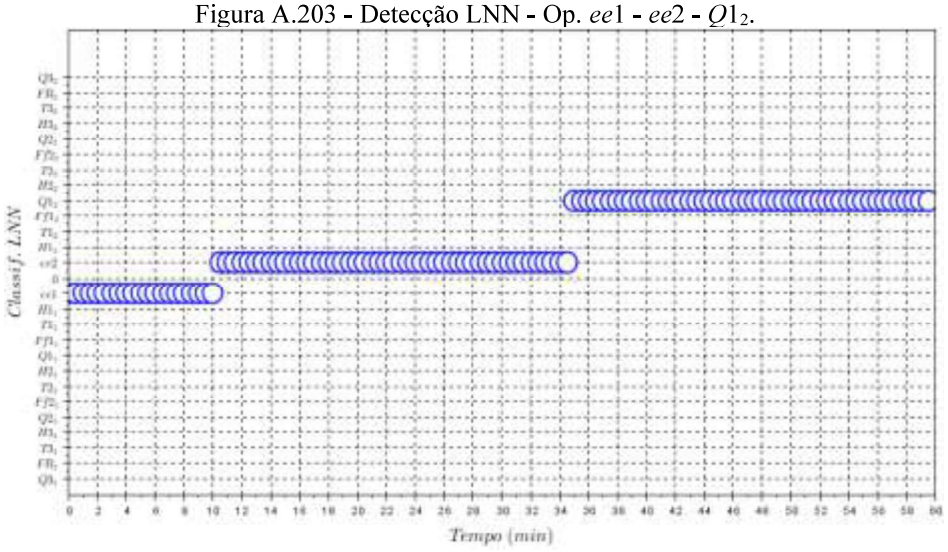


Figura A.204 - Detecção LNN - Op. *ee*2 - *ee*1 - $H2_1$.

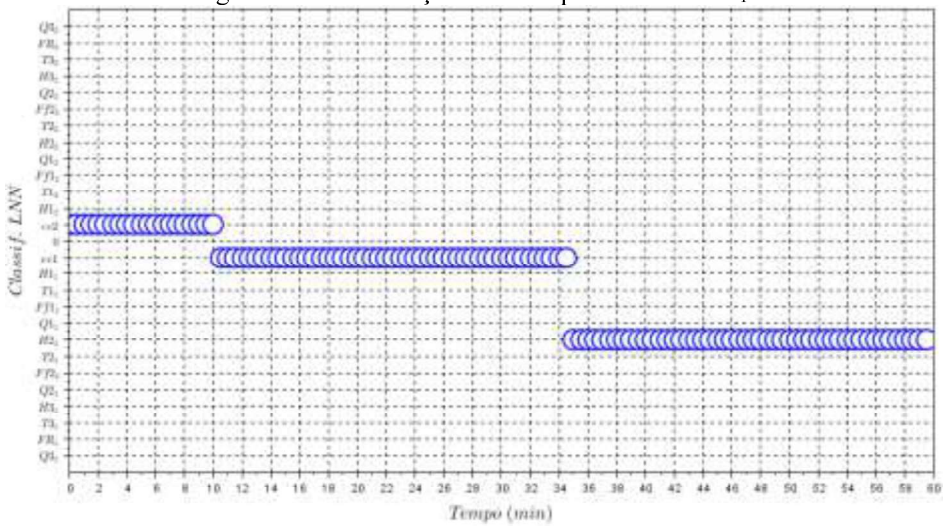


Figura A.205 - Detecção LNN - Op. *ee*1 - *ee*2 - $H2_2$.

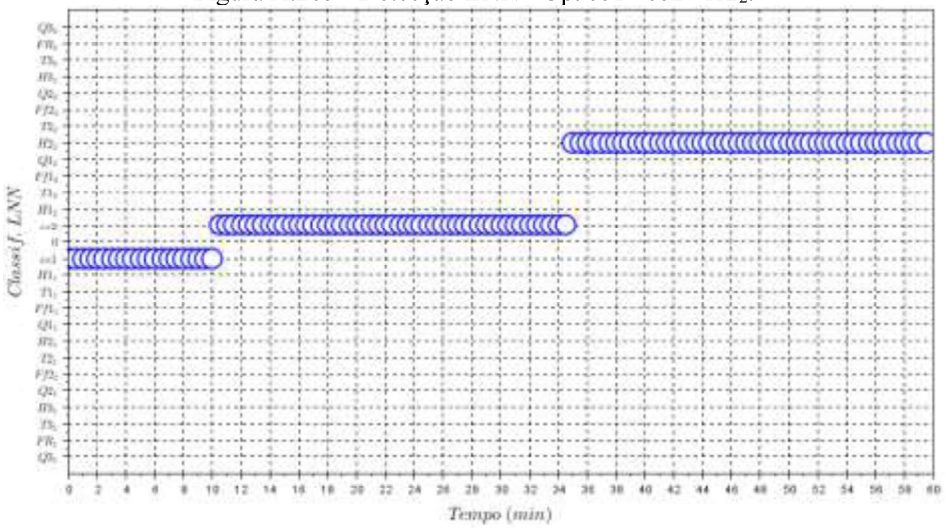


Figura A.206 - Detecção LNN - Op. *ee*2 - *ee*1 - $T2_1$.

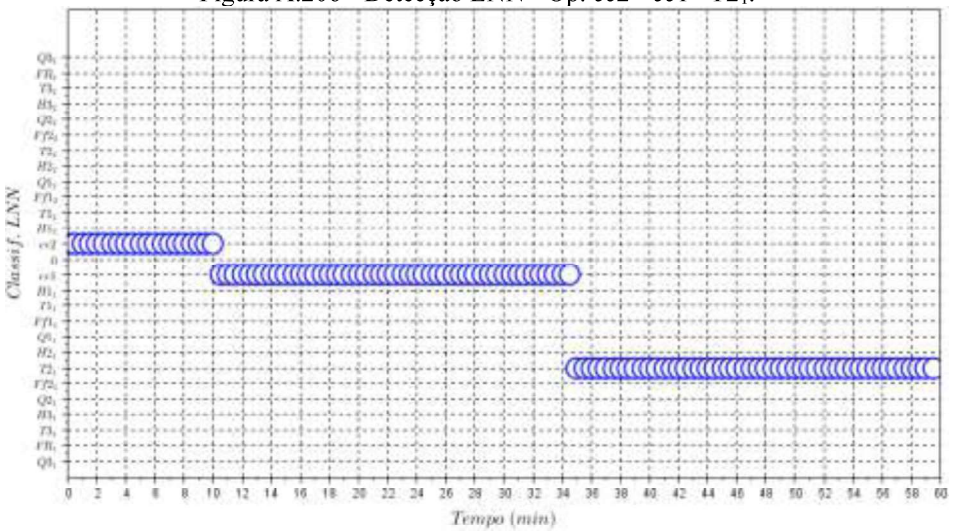


Figura A.207 - Detecção LNN - Op. *ee*1 - *ee*2 - $T2_2$.

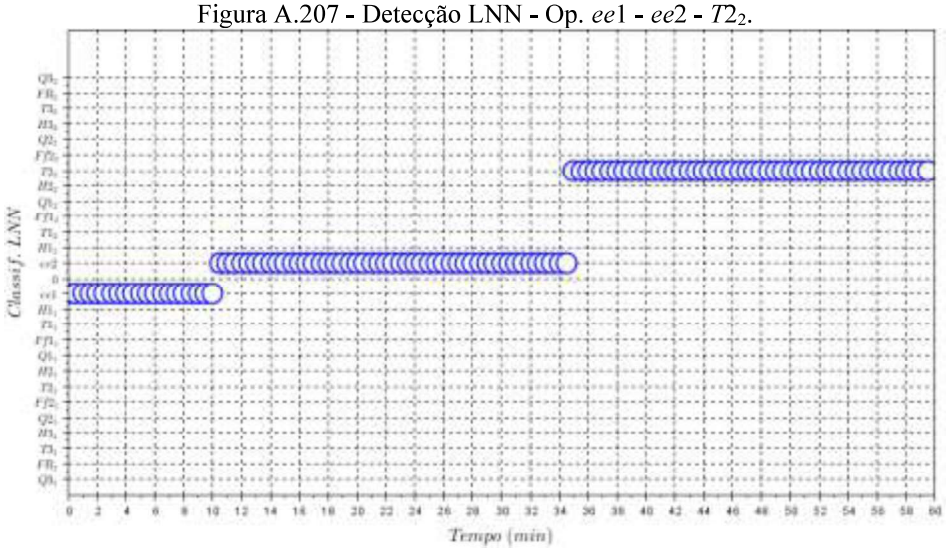


Figura A.208 - Detecção LNN - Op. *ee*2 - *ee*1 - $Ff2_1$.

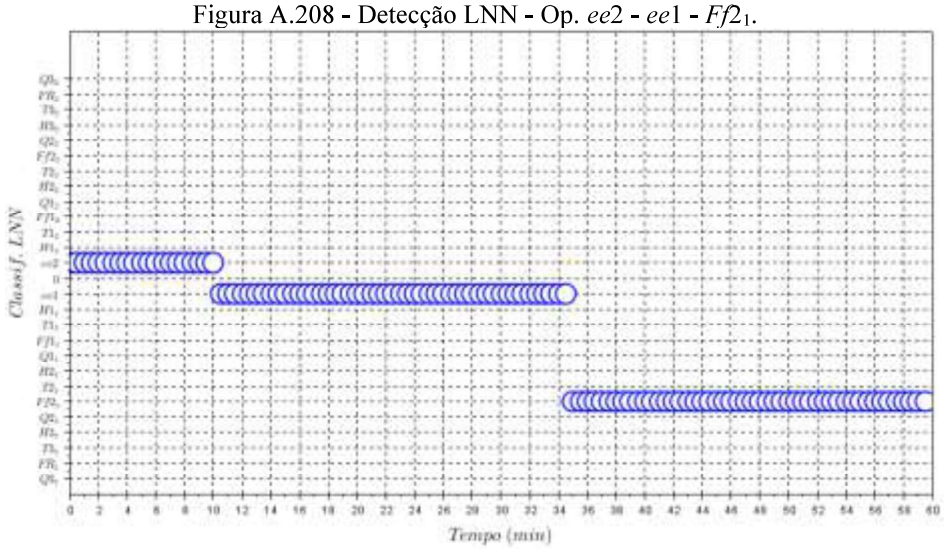


Figura A.209 - Detecção LNN - Op. *ee*1 - *ee*2 - $Ff2_2$.

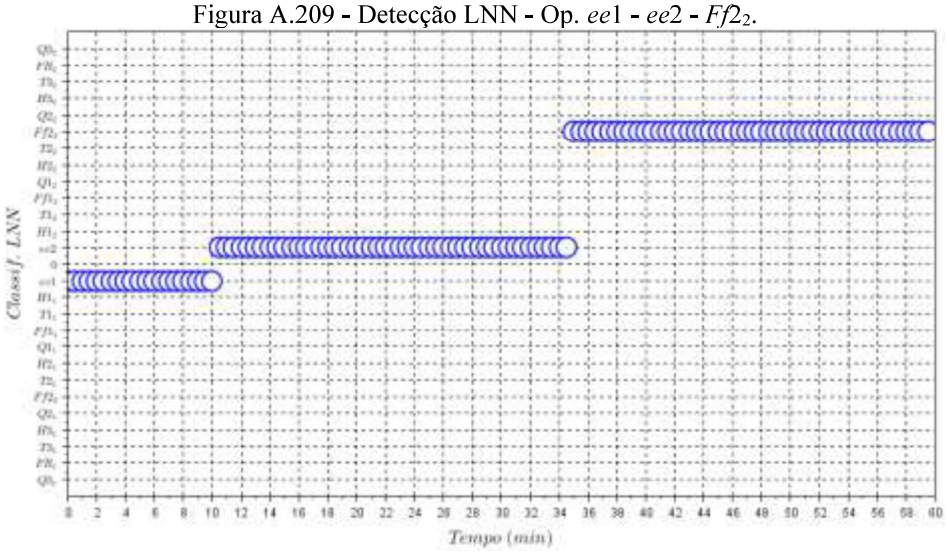


Figura A.210 - Detecção LNN - Op. *ee*2 - *ee*1 - $Q2_1$.

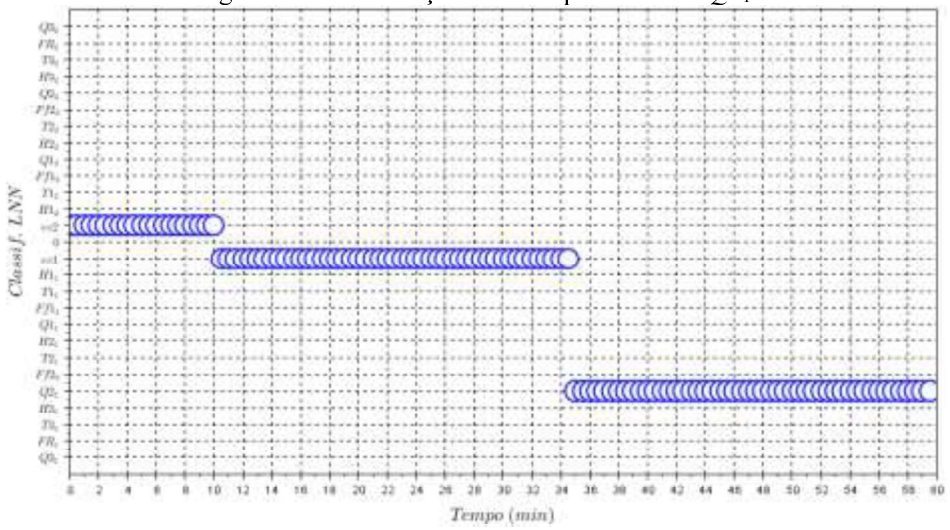


Figura A.211 - Detecção LNN - Op. *ee*1 - *ee*2 - $Q2_2$.

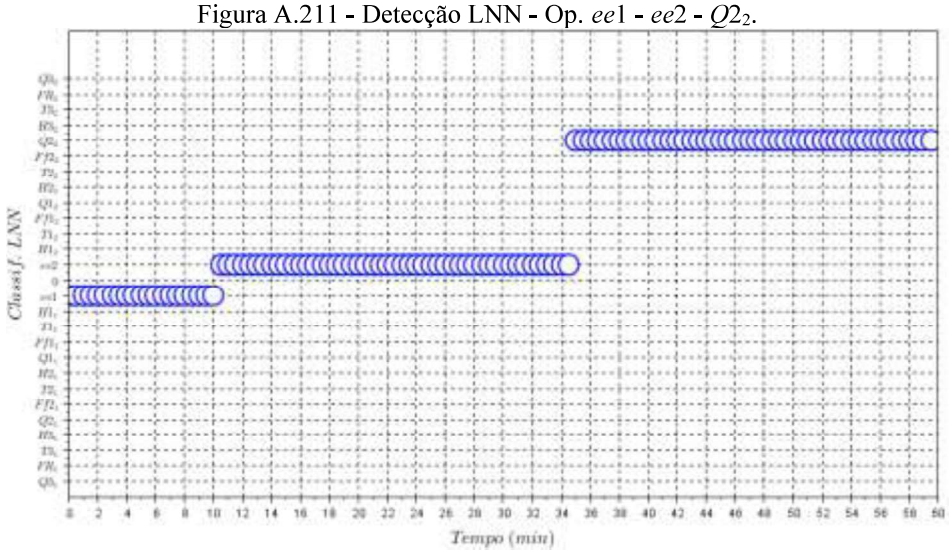


Figura A.212 - Detecção LNN - Op. *ee*2 - *ee*1 - $H3_1$.

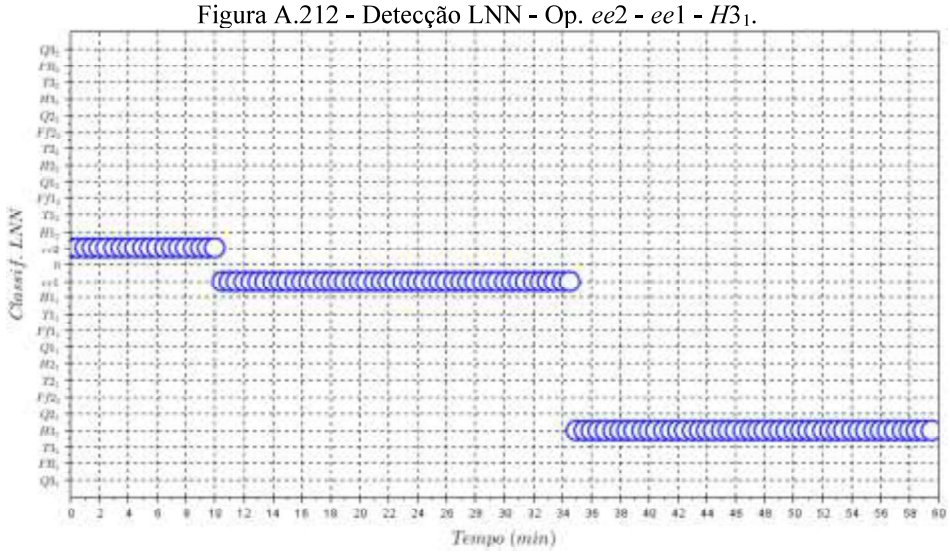


Figura A.213 - Detecção LNN - Op. *ee*1 - *ee*2 - $H3_2$.

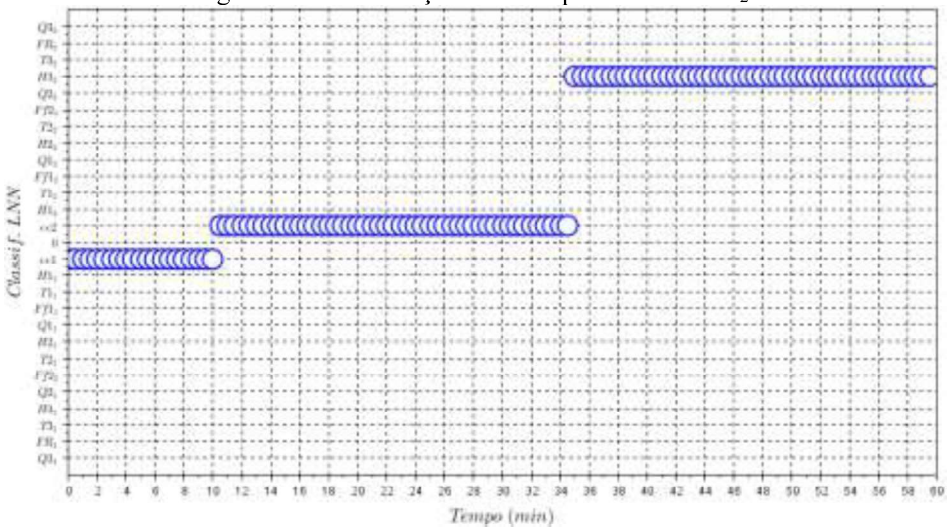


Figura A.214 - Detecção LNN - Op. *ee*2 - *ee*1 - $T3_1$.

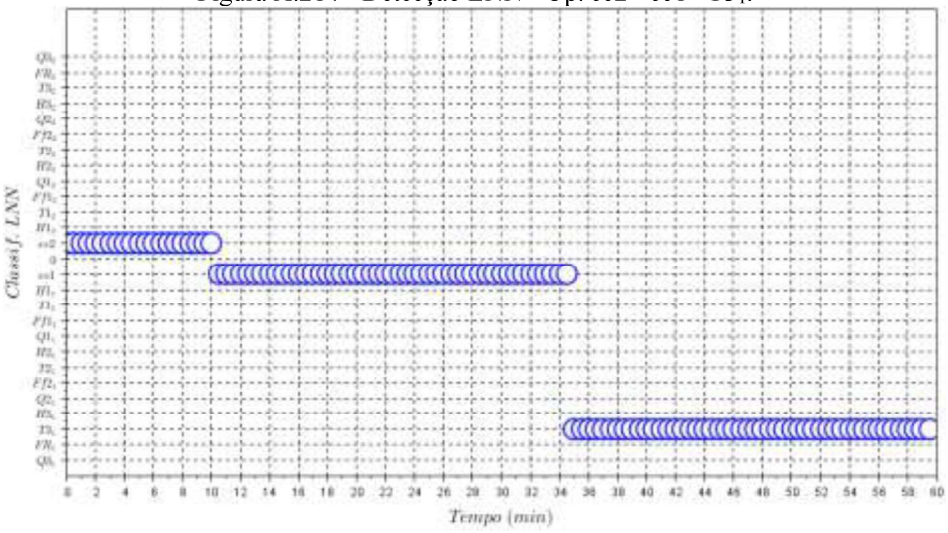


Figura A.215 - Detecção LNN - Op. *ee*1 - *ee*2 - $T3_2$.

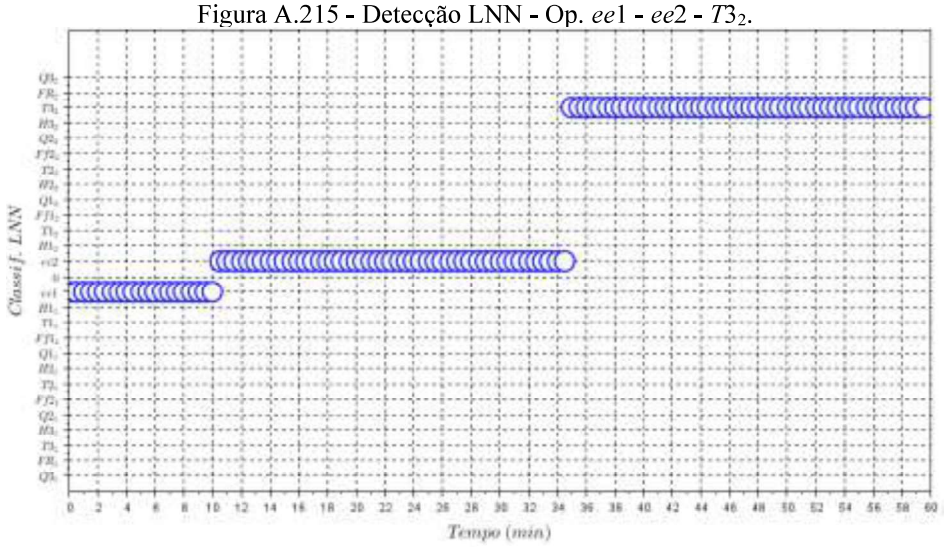


Figura A.216 - Detecção LNN - Op. *ee*2 - *ee*1 - F_{R1}.

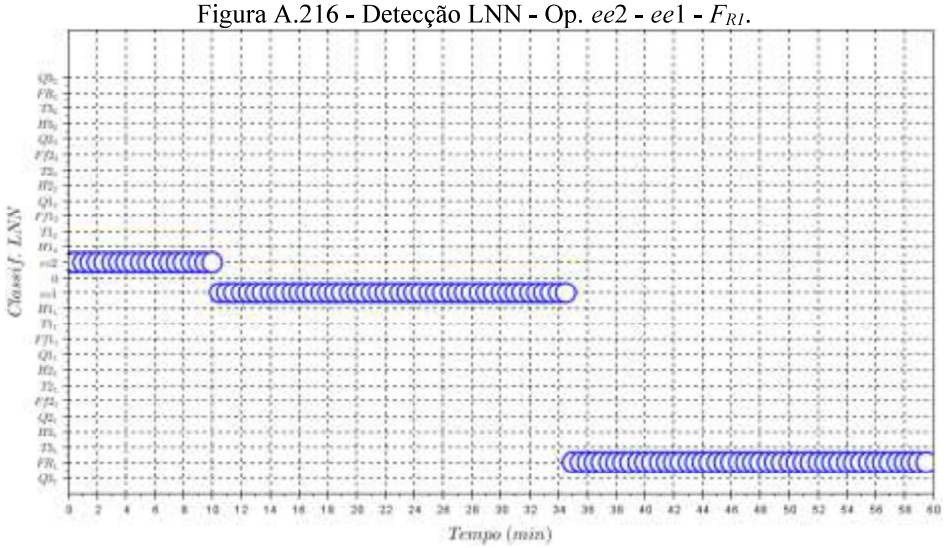


Figura A.217 - Detecção LNN - Op. *ee*1 - *ee*2 - F_{R2}.

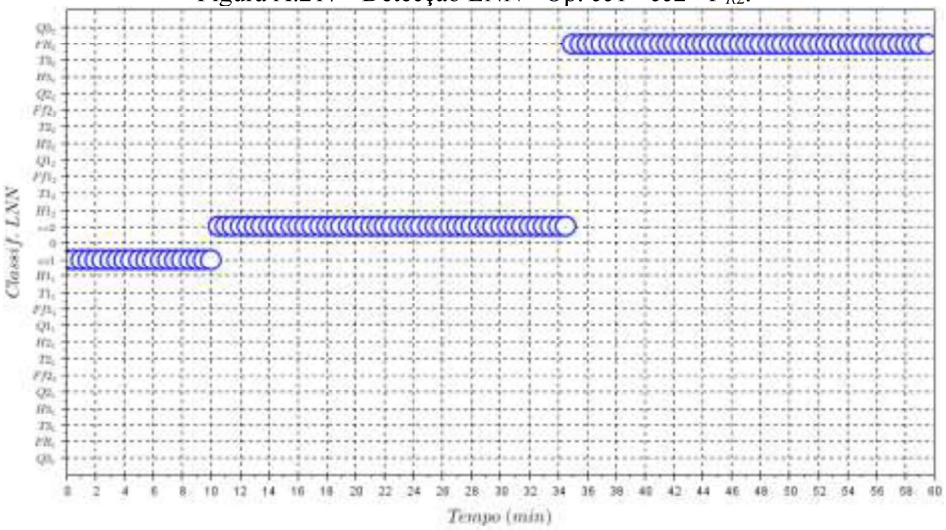


Figura A.218 - Detecção LNN - Op. *ee*2 - *ee*1 - $Q3_1$.

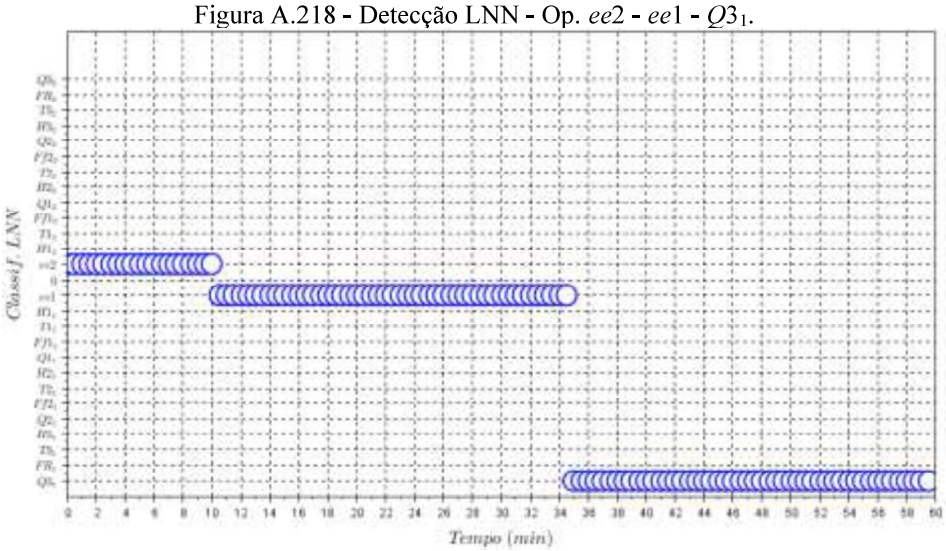
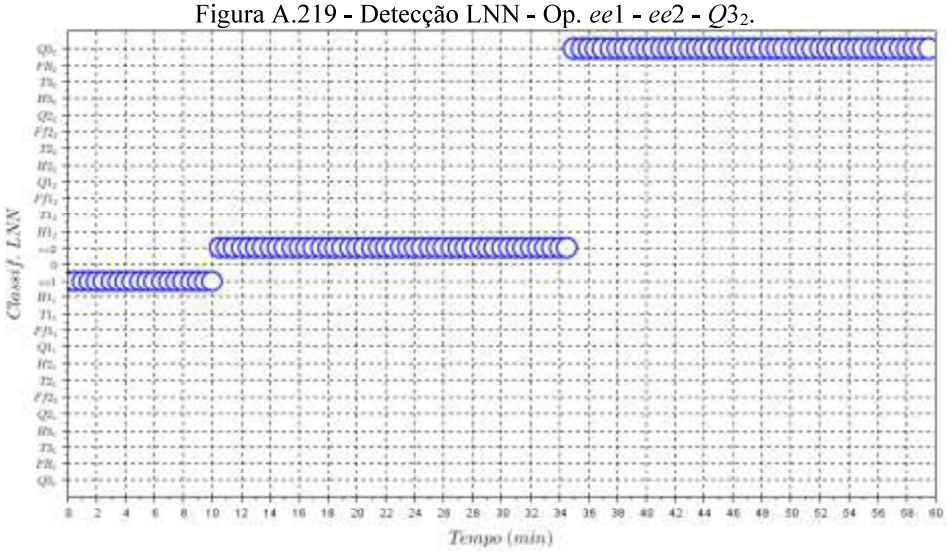


Figura A.219 - Detecção LNN - Op. *ee*1 - *ee*2 - $Q3_2$.



Os resultados obtidos com métodos de detecção de falhas com aprendizado profundo foram muito próximos dos resultados obtidos com redes neuronais, máquinas de vetores de suporte e lógica difusa. Assim, a metodologia de detecção de falhas através de aprendizado profundo apresenta a capacidade de realizar o propósito para que foi desenvolvida, porém com vantagens quanto a preparação dos dados de entrada, tornando o método útil para sistemas cujos dados não são completamente conhecidos e rotulados.