



UNIVERSIDADE FEDERAL DE UBERLÂNDIA
FACULDADE DE ENGENHARIA ELÉTRICA
ENGENHARIA ELETRÔNICA E DE TELECOMUNICAÇÕES

KREISLER BRENNER MENDES

COMPARATIVO DE ALGORITMOS CLÁSSICOS DE APRENDIZADO DE
MÁQUINA EM UM PROBLEMA DE RECONHECIMENTO DE FACES

PATOS DE MINAS - MG
2019

KREISLER BRENNER MENDES

COMPARATIVO DE ALGORITMOS CLÁSSICOS DE
APRENDIZADO DE MÁQUINA EM UM PROBLEMA DE
RECONHECIMENTO DE FACES

Trabalho apresentado como um dos requisitos parciais para aprovação na disciplina de Trabalho de Conclusão de Curso II do curso de Engenharia Eletrônica e de Telecomunicações da Universidade Federal de Uberlândia - Campus Patos de Minas.

Orientadora: Prof.^a Dr.^a Eliana Pantaleão.

PATOS DE MINAS - MG
2019

KREISLER BRENNER MENDES

COMPARATIVO DE ALGORITMOS CLÁSSICOS DE
APRENDIZADO DE MÁQUINA EM UM PROBLEMA DE
RECONHECIMENTO DE FACES

Trabalho apresentado como um dos requisitos
parciais para aprovação na disciplina de Trabalho de
Conclusão de Curso II do curso de Engenharia
Eletrônica e de Telecomunicações da Universidade
Federal de Uberlândia - Campus Patos de Minas.

Patos de Minas, 06 de dezembro de 2019.

COMISSÃO EXAMINADORA:

Prof.^a Dr.^a Eliana Pantaleão – FACOM/UFU

(Orientadora)

Prof. Dr. Laurence Rodrigues do Amaral – FACOM/UFU

(Examinador)

Prof. Dr. Pedro Luiz Lima Bertarini – FEELT/UFU

(Examinador)

AGRADECIMENTOS

Agradeço primeiramente a Deus, por me guiar e me dar forças para persistir e acreditar que tudo é possível quando se tem fé.

Aos meus pais Antonio Celmo Mendes e Nailde Mendes Luz que me concederam a melhor educação e lutaram para que eu concluísse mais essa etapa da minha vida. Ao meu irmão por todo companheirismo. A toda minha família e amigos por toda forma de apoio fornecida.

Agradeço especialmente a minha orientadora Prof^a. Dr^a Eliana Pantaleão por todo apoio, tempo dedicado, ensinamentos e paciência, não somente neste trabalho, mas ao longo de toda a minha graduação.

Aos professores Dr. Pedro Luiz Lima Bertarini e Dr. Laurence Rodrigues do Amaral cuja dedicação e atenção foram essenciais para o aperfeiçoamento desse trabalho.

A todo o corpo docente que contribuiu diretamente na minha formação profissional, por todo conhecimento doado, motivação e desafios que ajudaram no meu desenvolvimento.

E por fim agradeço também a Universidade Federal de Uberlândia, por toda a estrutura dada a minha educação e formação, como também por ter feito o intermédio de diversas amizades, que acredito que levarei para vida inteira.

“O óbvio é aquilo que ninguém enxerga, até que alguém o expresse com simplicidade”

— Kahlil Gibran

RESUMO

Com o aumento da disponibilidade de dados de faces, o aprendizado de máquina se torna uma ferramenta bastante relevante para a identificação de pessoas em diversas aplicações. Este trabalho busca avaliar o uso de algoritmos tradicionais de aprendizado de máquina sobre características extraídas de imagens de faces, com o intuito de realizar seu reconhecimento. As características escolhidas são baseadas na análise de componentes principais (PCA) que representam cada face como um vetor n -dimensional, sendo n menor que o tamanho da imagem em si. Os algoritmos avaliados são classificadores por distância, árvores de decisão, máquinas de vetores de suporte e redes neurais artificiais, que foram avaliados em suas diversas parametrizações. Os melhores resultados quanto à acurácia foram obtidos pelos modelos de máquina de vetores de suporte e redes neurais artificiais (99%), sendo que o último apresentou maior acurácia (97%) quando o mesmo modelo foi testado em um novo banco de dados.

Palavras-Chave: Reconhecimento Facial, Aprendizagem de Máquina, Análise de Componentes Principais, Árvores de Decisão, Máquina de Vetores de Suporte, Rede Neural Artificial.

ABSTRACT

With the increased availability of face data, machine learning becomes a very relevant tool for identify people in various applications. This work aims to evaluate the use of traditional machine learning algorithms on features extracted from face images, in order to perform their recognition. The extracted features are based on the Principal Component Analysis (PCA), that represent each face as a n -dimensional vector, where n is less than the image size. The evaluated algorithms are distance classifiers, decision trees, support vector machines and neural networks, which were evaluated in their different parametrizations. The best accuracy results were obtained by the support vector machine and artificial neural networks models (99%), with the latter presenting the highest accuracy (97%) when the same model was tested in a new database.

Keywords: Face Recognition, Machine Learning, Principal Component Analysis, Decision Tree, Support Vector Machine, Artificial Neural Network.

Lista de Figuras

Figura 1 - Etapas de um sistema de reconhecimento facial.....	17
Figura 2 - Comparação entre a imagem original (a) e a mesma com histograma equalizado (b).	19
Figura 3 - Análise de componentes principais em 2D.....	20
Figura 4 - Representação simplificada do espaço de faces.....	21
Figura 5- Arquitetura de uma árvore de decisão e as regiões de decisão no espaço de objetos.....	23
Figura 6 - Separação de duas classes por (a) três hiperplanos possíveis. (b) Hiperplano ótimo	25
Figura 7- Representação do hiperplano de separação para padrões não linearmente separáveis.....	26
Figura 8 - Exemplo de transformação realizada em um conjunto de dados não linear para um novo espaço de características de maior dimensão.....	27
Figura 9 - Exemplo de aplicação de técnicas multiclasse OAA e OAO em um problema de separação envolvendo 3 classes	28
Figura 10 - Diagrama de blocos de um neurônio artificial.....	29
Figura 11 - Exemplo de RNA multicamada típica	30
Figura 12 - Exemplo de uma matriz de confusão para um problema de 3 classes	31
Figura 13 - Etapas do desenvolvimento deste trabalho	33
Figura 14 - Segmentação de face.....	34
Figura 15 – Fluxo dos testes realizados para a comparação dos classificadores.....	36
Figura 16 - Comparação de imagens do banco de faces (a) antes e (b) após o recorte e equalização do histograma.....	41
Figura 17 - Relação do número de componentes com a variância da base de dados Yale Face Database	42
Figura 18 - Acurácia dos modelos de DT obtidos pela variação dos parâmetros Min_Amostras_Folha e Max_Profundidade com critério Entropia	45
Figura 19 - Acurácia dos modelos de DT obtidos pela variação dos parâmetros Min_Amostras_Folha e Max_Profundidade com critério Gini	45
Figura 20 – Taxas de Acurácia dos modelos SVM obtidas pelas variações do parâmetro C do <i>kernel</i> Linear	48

Figura 21 - Acurácia dos modelos SVM com a variação dos parâmetros C e Γ do <i>kernel</i> RBF	49
Figura 22 - Acurácia dos modelos SVM com a variação dos parâmetros C e Γ do <i>kernel</i> Sigmoidal	50
Figura 23 - Acurácia dos modelos SVM com a variação dos parâmetros C , Γ e <i>Degree</i> do <i>kernel</i> Polinomial	51
Figura 24 - Acurácia de modelos RNA com uma HL em função do número de neurônios artificiais e AF.	55
Figura 25 - Acurácia dos modelos de RNAs com duas HLs a partir da variação da quantidade de neurônios artificiais de cada HL utilizando AF ReLU.....	56
Figura 26- Acurácia dos modelos de RNAs com duas HLs a partir da variação da quantidade de neurônios artificiais de cada HL utilizando AF Identidade	56
Figura 27- Acurácia dos modelos de RNAs com duas HLs a partir da variação da quantidade de neurônios artificiais de cada HL utilizando AF Logística..	57
Figura 28- Acurácia dos modelos de RNAs com duas HLs a partir da variação da quantidade de neurônios artificiais de cada HL utilizando AF Tanh	57
Figura 29 - Matriz de confusão do melhor classificador de RNA e SVM, ambas matrizes foram idênticas.....	60
Figura 30 - Comparação das classes (a) verdadeira e (b) predita erroneamente pelos classificadores SVM e RNA	61
Figura 31 - Acurácia dos modelos de SVM obtidos com a variação do parâmetro C do <i>kernel</i> Linear para os bancos de faces <i>Yale Face Database</i> e <i>ORL Dataset of Faces</i>	63
Figura 32 - Acurácia dos modelos de RNAs com uma HL a partir da variação da quantidade de neurônios artificiais utilizando AF Logística com imagens dos bancos de dados <i>Yale Face Database</i> e <i>ORL Dataset of Faces</i>	63

Lista de Quadros

Quadro 1 - Exemplo de funções de <i>kernel</i> e seus respectivos parâmetros	27
Quadro 2 - Bancos de imagens faciais.....	39
Quadro 3 - Resultados do experimento utilizando métricas de similaridade	44
Quadro 4 - Melhores modelos obtidos com DT para cada critério utilizado.....	46
Quadro 5 - Resultado médio de todos os modelos DT testados com cada critério utilizado	46
Quadro 6 - Melhor modelo do método SVM utilizando <i>kernel</i> Linear	48
Quadro 7 - Melhor modelo do método SVM utilizando <i>kernel</i> RBF	50
Quadro 8 - Melhor modelo do método SVM utilizando <i>kernel</i> Sigmoidal	50
Quadro 9- Melhor modelo do método SVM utilizando <i>kernel</i> Polinomial	52
Quadro 10 – Resultado médio de todos os modelos SVM testados com cada função de <i>kernel</i> testado	52
Quadro 11 - Melhores modelos de RNA obtidos com uma HL	55
Quadro 12 - Melhores modelos de RNA obtidos com duas HLs	58
Quadro 13 - Melhores modelos de RNA obtidos com três HLs.....	58
Quadro 14 - Resultado médio de todos os modelos de RNAs testados com cada combinação de AFs com a quantidade de HLs.....	59
Quadro 15 - Resultados obtidos nos testes com o melhor classificador SVM utilizando o banco de faces <i>ORL Database of faces</i>	62
Quadro 16 - Resultados obtidos nos testes com o melhor classificador RNA utilizando o banco de faces <i>ORL Database of faces</i>	62

Lista de Abreviações e Siglas

ANN	<i>Artificial Neural Network</i>	Rede Neural Artificial
AF	<i>Activation Function</i>	Função de Ativação
DT	<i>Decision Tree</i>	Árvore de Decisão
FR	<i>Face Recognition</i>	Reconhecimento Facial
HL	<i>Hidden Layer</i>	Camada Oculta
k-NN	<i>k-Nearest Neighbor</i>	k-Vizinhos mais próximos
ML	<i>Machine Learning</i>	Aprendizado de Máquina
MLP	<i>Multilayer Perceptron</i>	Perceptron Multicamadas
OAA	<i>One-Against-All</i>	Um-Contra-Todos
OAo	<i>One-Against-One</i>	Um-Contra-Um
PCA	<i>Principal Component Analysis</i>	Análise de Componentes Principais
RBF	<i>Radial Basis Function</i>	Função de Base Radial
ReLU	<i>Rectified Linear Unit</i>	Unidade Linear Retificada
SVM	<i>Support Vector Machine</i>	Máquina de Vetores de Suporte

Sumário

Capítulo 1.....	13
Introdução	13
1.1 Tema do Projeto	14
1.2 Problematização	14
1.3 Objetivos	14
1.3.1 Objetivos Gerais	14
1.3.2 Objetivos Específicos	15
1.4 Justificativas	15
1.5 Considerações Finais	15
Capítulo 2.....	16
Desenvolvimento Teórico.....	16
2.1 Biometria	16
2.2 Detecção Facial	17
2.3 Normalização	18
2.3.1 Equalização de Histograma	18
2.4 Extração de características	19
2.5 PCA	20
2.6 Métricas de Similaridade	21
2.6.1 Distância Euclidiana	21
2.6.2 Distância Manhattan	22
2.6.3 Distância Chebyshev	22
2.6.4 Distância de Mahalanobis	22
2.7 Árvore de Decisão (DT)	22
2.8 Máquina de Vetores de Suporte (SVM)	24
2.8.1 SVM de margem suave	25
2.8.2 SVM não-linear	26
2.8.3 SVM multiclasse	27
2.9 Redes Neurais Artificiais	28
2.10 Avaliação do Algoritmo	30
2.10.1 Métricas de Classificação	30
2.10.2 Matriz de Confusão	31
2.10.3 Validação Cruzada	32
Capítulo 3.....	33
Metodologia	33
3.1 ETAPA I: Preparação das amostras	34
3.2 ETAPA II: Treinamento	34
3.3 ETAPA III: Teste	35
3.4 ETAPA IV: Comparação de Resultados	35
Capítulo 4.....	37
Recursos.....	37
4.1 Linguagem Python	37
4.2 Bibliotecas	37
4.3 Máquina utilizada para realização dos testes	37
4.4 Banco de dados de imagens faciais	37
Capítulo 5.....	40
Experimentos e Resultados	40
5.1 Segmentação e Equalização de Histograma	40
5.2 Redução de dimensionalidade com PCA	41

5.3	Validação cruzada utilizando a biblioteca <i>scikit-learn</i>	43
5.4	Experimentos utilizando métricas de similaridade	43
5.5	Experimentos utilizando DT	44
5.6	Experimentos utilizando SVM.....	47
5.7	Experimentos utilizando RNA.....	53
5.8	Comparação dos melhores modelos de cada técnica utilizada	59
Capítulo 6.....		64
Considerações Finais		64
Referências		66

Capítulo 1

Introdução

Nos últimos anos tem-se observado um constante aumento da qualidade e número de dispositivos com câmera integrada no mercado. Desta forma, pesquisas relacionadas a aprendizado de máquina (ML do inglês *machine learning*) estão em alta, devido ao grande volume de informações geradas por estes dispositivos. Sistemas de ML estão sendo providenciais para encontrar padrões em diversas bases de dados, realizando uma coleta de informações lapidadas fazendo com que aplicações se tornem inteligentes. Esses sistemas podem trazer soluções desde problemas simples a problemas que um ser humano não seria capaz de solucionar [1].

Pode-se subdividir a aprendizagem de máquina em dois tipos principais: supervisionado e não-supervisionado. No supervisionado, o processo de aprendizagem é desenvolvido à partir de uma base de dados rotulada. Por exemplo, em um sistema, a tarefa é avaliar se uma imagem contém bananas ou maçãs. O aprendizado será iniciado a partir um banco de imagens de bananas e maçãs, e cada imagem do banco deve conter a informação que indica qual fruta a imagem representa. Desta forma, ao se entrar com uma imagem de uma das frutas sem rótulos (sem informação de qual fruta a imagem representa) nesse sistema, será possível interpretá-la e classificá-la como banana ou maçã. Já no processo não-supervisionado, diferentemente do supervisionado, não existem dados rotulados, ou seja, cada imagem contida na base de dados não possui a informação que descreve o seu conteúdo. Um processo não-supervisionado para o exemplo anterior teria uma finalidade diferente, pois ao se utilizar o mesmo banco de imagens de bananas e maçãs (sem rótulos), seria possível identificar a existência de dois frutos distintos, mas não caberia ao sistema identificar qual o fruto contido na imagem [2]. Existe ainda a abordagem semi-supervisionada, que usa tanto dados rotulados quanto dados sem rótulos. Essa abordagem é comum quando há dificuldade em conseguir grande quantidade de dados rotulados em uma aplicação. Os dados não-rotulados são usados para inferir os agrupamentos das amostras, enquanto os poucos dados rotulados fornecem a identificação da classe dos elementos.

Dentro do universo de ML várias tecnologias vêm sendo desenvolvidas e/ou otimizadas. Dentre elas temos a tecnologia de Reconhecimento Facial (FR, do inglês *Face Recognition*), que surgiu como uma útil ferramenta para reconhecer características de rostos por meio de seus

traços inerentes. Essa tem sido uma das áreas mais pesquisadas no campo do reconhecimento de padrões e visão computacional. Esse interesse é impulsionado pelas aplicações em áreas como segurança, vigilância, controle de acesso, software de análise de vídeo e milhares de aplicativos em nossas vidas diárias, como entretenimento, compras inteligentes e identificação automática de rostos em coleções de fotos [3].

Os seres humanos são capazes de reconhecer centenas de rostos, aprendendo ao longo de toda a sua vida e identificar e reconhecer rostos familiares facilmente, mesmo após a separação de alguns anos. No entanto, o cérebro humano se limita ao número total de pessoas que se pode lembrar com precisão, seria em média 5000 faces [4]. Uma das principais vantagens de um sistema de computador é sua capacidade de lidar com grandes números de imagens faciais [5].

1.1 Tema do Projeto

Um sistema de FR tem como meta identificar uma pessoa por meio de imagem ou vídeo. Seu uso se tornou mais perceptível e acessível nos últimos anos e pode ser usado em diversas áreas de segurança, biometria, e em soluções inovadoras, como aplicativos pessoais de foto e autenticação secundária para dispositivos móveis.

1.2 Problematização

Atualmente existem diversas técnicas de FR que atingem uma taxa bastante significativa de acertos, entretanto existem fatores que prejudicam a acurácia desses sistemas. Esses fatores são divididos em dois tipos: fatores intrínsecos e extrínsecos. Os fatores intrínsecos estão relacionados à aparência da face, por exemplo, envelhecimento, expressões faciais, etc., enquanto fatores extrínsecos são aqueles que se tornam uma razão para mudar a aparência da face, por exemplo: condição de iluminação, variação de pose, obstrução da face, baixa resolução da imagem e faces similares de pessoas diferentes [6].

1.3 Objetivos

1.3.1 Objetivos Gerais

O objetivo deste trabalho é realizar um estudo comparativo de classificadores e suas configurações aplicados ao FR utilizando a técnica de Análise Componentes Principais (PCA) para obtenção do vetor de características.

1.3.2 Objetivos Específicos

- Pesquisar, estudar, compreender e implementar o processo de FR para um específico banco de dados de faces.
- Calcular as métricas de desempenho dos algoritmos testados por meio dos métodos de avaliação definidos.
- Comparar o desempenho do FR em vários cenários obtidos com a variação de parâmetros dos métodos descritores e classificadores.

1.4 Justificativas

O tema FR vem sendo cada vez mais pronunciado, o que tem despertado a curiosidade de muitas pessoas quanto ao funcionamento desta tecnologia. Desta forma se faz necessária a investigação e o estudo de algumas técnicas utilizadas e o quão eficiente elas são, verificando também os desafios e dificuldades que devem ser superados. Além disso, as técnicas estudadas podem ser aplicadas em outros tipos de sistemas envolvendo aprendizado de máquinas.

1.5 Considerações Finais

Por meio deste trabalho, será possível compreender o contexto atual dos sistemas de FR, algumas aplicações, vantagens de utilização, além de comparar e analisar diferentes variações dos algoritmos baseados no descritor de características PCA, também permitirá compreender cada parte do processo de FR.

Este trabalho está organizado da seguinte forma: O Capítulo 2 contém o referencial teórico, no qual está descrito como funciona um processo de FR. Também são abordados alguns dos métodos de extração e classificação de características utilizados, juntamente com os métodos de avaliação dos algoritmos. No Capítulo 3 é apresentada a metodologia empregada neste trabalho, a qual é dividida em quatro etapas: (I) preparação das amostras, (II) treinamento, (III) teste, (IV) comparação dos resultados. No Capítulo 4 são descritos os recursos utilizados para o desenvolvimento deste trabalho. No Capítulo 5 são explicados detalhadamente todos experimentos realizados e seus respectivos resultados. No Capítulo 6 são apresentadas as considerações finais.

Capítulo 2

Desenvolvimento Teórico

Neste capítulo são apresentados os principais conceitos relacionados com as técnicas usadas na produção deste trabalho. São descritos conceitos de biometria, técnica de detecção facial, normalização das faces, extração de características, métricas de similaridade e métodos de classificação e de avaliação dos algoritmos.

2.1 Biometria

Biometria é a ciência que estabelece a autenticidade de um indivíduo baseada em suas características fisiológicas ou comportamentais. A tecnologia biométrica é um conjunto de técnicas aplicadas para capturar, analisar e identificar pessoas por meio de seus traços biométricos em representação digital [7]. Traços biométricos são características singulares de cada indivíduo utilizadas para distingui-lo de outros indivíduos. A relevância da biometria na sociedade moderna foi reforçada pela necessidade de sistemas de gerenciamento de identidade em grande escala cuja funcionalidade depende da determinação precisa da identidade de um indivíduo no contexto de várias aplicações diferentes [8]. A grande vantagem da utilização de tecnologia biométrica é que não existe o risco de perda ou esquecimento de dados autenticadores, assim como ocorre com senhas, *tokens* e documentos [9]. Por outro lado, as técnicas biométricas podem apresentar problemas, tais como, demora na autenticação, biometria comprometida, falsas detecções, permitindo acesso a quem não deveria ter, ou o contrário, não permitindo acesso a quem deveria ter. Estes problemas podem ser evitados dependendo da tecnologia biométrica empregada.

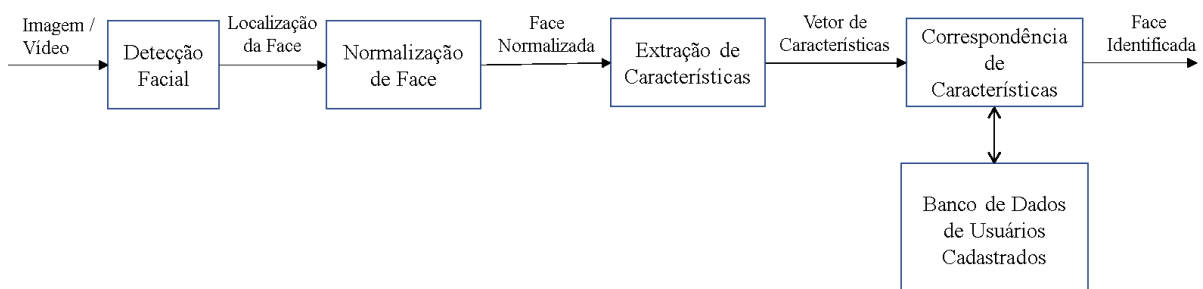
Dentre as tecnologias biométricas encontra-se a biométrica por FR que consiste em identificar padrões que distinguem os indivíduos tais como distância entre os olhos, formato da boca entre outros [7]. O FR é uma tarefa que normalmente é trivial para seres humanos, já para uma máquina pode conter diversos processos e inúmeros cálculos. O FR oferece algumas vantagens sobre outras tecnologias biométricas. A primeira delas é que não é necessária uma participação ativa do usuário a ser identificado assim como acontece em biometria por reconhecimento de digital, palma da mão ou íris em que o usuário precisar manter um contato físico ou estar bastante próximo do dispositivo leitor. Outra vantagem é que pode ser realizado o reconhecimento de vários usuários ao mesmo tempo, muito aproveitada em aplicações de segurança [10]. As desvantagens desse tipo de biometria para com a maioria das biometrias

seriam quanto ao desempenho, devido ao maior volume de processamento, e maior vulnerabilidade à fraude [11].

Um sistema de FR geralmente consiste em quatro módulos, como mostrado na Figura 1: (1) detecção, (2) normalização, (3) extração de características e (4) correspondência. Detecção e normalização são etapas de pré-processamento. A detecção facial segmenta as áreas da face do plano de fundo. A normalização ajuda a corrigir fatores que dificultam o reconhecimento, tal como a iluminação do ambiente [12].

Após a normalização, a extração de características é executada para fornecer informações efetivas que são úteis para distinguir entre faces de pessoas diferentes. O ideal é que este processo seja estável em relação às variações geométricas e fotométricas. Para classificação de face, o vetor de características extraído da face de entrada é comparado com aqueles das faces registradas no banco de dados. O sistema fornece a identidade da face quando uma correspondência for encontrada com o nível de confiança suficiente, caso contrário indica uma face como desconhecida [12].

Figura 1 - Etapas de um sistema de reconhecimento facial



Fonte: Modificada de [10]

2.2 Detecção Facial

A detecção de rosto é o primeiro passo no RF. Sua precisão tem um papel importante no desempenho e na usabilidade de todo o sistema de RF. Dada uma única imagem ou um vídeo, um detector facial ideal deve ser capaz de identificar e localizar todas as faces presentes, independentemente de sua posição, escala, orientação, idade e expressão. Além disso, a detecção deve ser independente das condições de iluminação externas e do conteúdo da imagem e do vídeo [10]. Uma técnica de detecção facial comumente empregada em ambientes controlados é baseada nas características de Haar descrito em [13].

2.3 Normalização

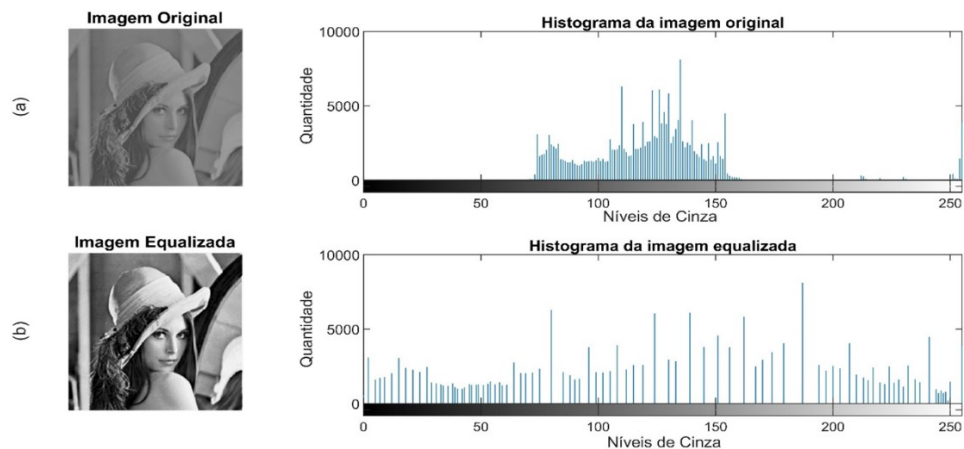
O objetivo da normalização de imagens é remover efeitos ou sinais que não tenham relevância e que apresentem singularidades não desejadas intrínsecas ao processo de aquisição das imagens [14]. A normalização supre variações existentes em uma imagem facial, atuando fortemente em aspectos como iluminação, aproximação e posição, reproduzindo ao final uma imagem o mais próxima possível do padrão do banco de dados da comparação [15].

Após a detecção de face em uma imagem normalmente inicia-se o processo de normalização, o qual, pode ser dividido em normalização geométrica e fotométrica. A normalização geométrica resume-se no alinhamento da altura dos olhos, nivelar a escala das faces e corrigir a posição das faces com relação a imagem. Já a normalização fotométrica consiste em reduzir ruídos, sombras e variações de contraste e/ou brilho da imagem a fim de melhorar o reconhecimento [16].

2.3.1 Equalização de Histograma

Muitas imagens de face são adquiridas sob condições de iluminação distintas fazendo com que amostras faciais de um mesmo indivíduo tenham um grande nível de distinção em termos de tons de cinza o que pode influenciar o desempenho do sistema de FR. A Equalização de Histograma tem como objetivo redistribuir os valores de tons de cinza dos *pixels* em uma imagem, de forma a obter um histograma mais homogêneo. Com o aumento do intervalo dinâmico entre os níveis de cinza, ocorre uma melhora no contraste em imagens adquiridas sob condições de iluminação não ideais, o que torna possível comparar cenas que foram adquiridas com disposições de iluminações distintas. Assim a equalização de histograma se torna um artifício importante no processo de normalização das imagens de face [14, 16]. Por meio da Figura 2 pode-se observar a diferença visual gerada pela equalização de histograma tanto da imagem em si quanto no gráfico do histograma correspondente. Na Figura 2 (a) tem-se a imagem original e à direita tem-se seu respectivo histograma e na Figura 2 (b) tem-se a imagem com seu histograma equalizado. Comparando os dois histogramas mencionados, nota-se o efeito do espalhamento das intensidades de cores geradas pela equalização.

Figura 2 - Comparação entre a imagem original (a) e a mesma com histograma equalizado (b).



Fonte: Modificada de: [17]

2.4 Extração de características

Um dos maiores problemas no reconhecimento de padrões é a dimensionalidade das características de um conjunto de dados. Grande parte dos problemas envolvem um número elevado de dados a serem processados, gerando um alto custo computacional. Uma forma de contornar esse problema é utilizando métodos de extração de características, que consistem em encontrar novas representações para os dados. Com isso, é possível reduzir ao mínimo o número de dimensões de um problema sem que haja perdas de informações relevantes [18]. “Devemos procurar selecionar recursos que levem a grandes distâncias entre classes e pequena variação dentro da classe no espaço vetorial de características” [18, p. 164].

Assim, em problemas no qual envolvem reconhecimento de padrões em imagens esses métodos passam a ser indispensáveis devido à grande quantidade de informação que uma imagem pode conter. Desta forma pode-se dispensar *pixels* redundantes a fim de reduzir as dimensões do problema. Existem inúmeros métodos de extração de características utilizados em FR, que podem ser classificados como holísticos, locais ou híbridos [5].

Os métodos holísticos são aqueles que procuram identificar faces usando representações globais da imagem facial. Nos métodos locais, características locais como olhos, nariz e boca são primeiramente extraídas e suas localizações e estatísticas locais (geométricas e/ou aparência) são alimentadas em um classificador local. Nos métodos híbridos, assim como o sistema de percepção humana usa os recursos locais e toda a região da face para reconhecer uma face, um sistema de reconhecimento de máquina deve usar ambos [5, 15].

2.5 PCA

A Análise de Componentes Principais, do inglês *Principal Component Analysis* (PCA), é um método estatístico de redução de dimensionalidade de um conjunto de dados de entrada mantendo apenas os dados mais relevantes. Para melhor entendimento do PCA considera-se o gráfico da Figura 3, onde os pontos representam um conjunto de dados qualquer distribuídos ao longo dos eixos X e Y.

Figura 3 - Análise de componentes principais em 2D



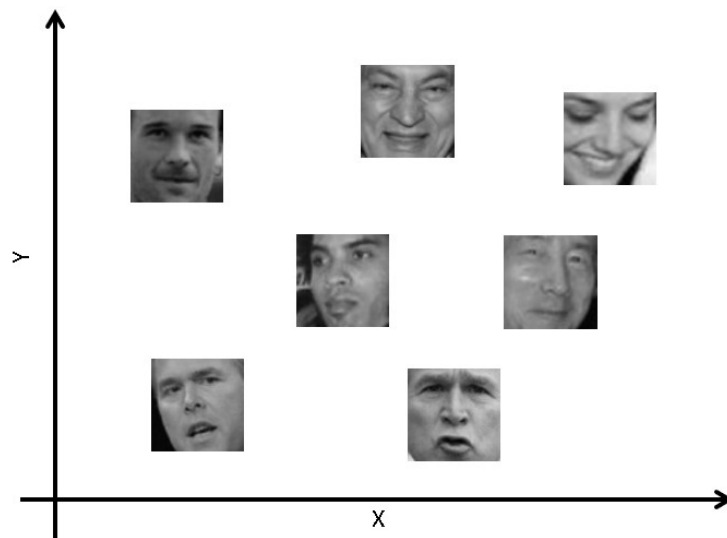
Fonte: Autor.

Observando a Figura 3 nota-se que a forma de distribuição dos dados é semelhante ao de uma elipse. Assim ao longo do eixo principal desta elipse encontra-se a direção de máxima variação dos dados, ou seja, a direção que contém mais informação, que é conhecida como primeira componente principal e está representada pela reta de maior comprimento. O segundo componente principal é a direção de maior variação dos dados perpendicular à primeira componente principal, representado pela reta de menor comprimento.

Assim podemos representar os dados por um novo sistema de coordenadas, onde a reta que possui maior comprimento no gráfico pode ser definida como uma dimensão X' e a reta perpendicular a X' como uma dimensão Y' . A fim de reduzir a quantidade de dados redundantes pode-se eliminar a dimensão Y' sem perdas relevantes. Pode parecer pouco a redução de dados em um exemplo em 2D, mas considerando um sistema em que possua 5000D e, por meio da utilização do método PCA, possa ser diminuído para apenas as 50D principais, pode-se reduzir bastante a quantidade de dados a serem utilizados no sistema em questão.

A técnica PCA é utilizada para extração de características de imagens faciais em que cada *pixel* de uma imagem facial é considerado um atributo e então associado a uma dimensão em um espaço de dados [10, 19]. Após a extração de características é gerado um espaço de faces em que cada face é localizada por um vetor de N dimensões. O valor de N é selecionado conforme a porcentagem de informação que se deseja manter na representação. As características selecionadas não representam necessariamente características físicas específicas. Na Figura 4 é mostrada uma representação em duas dimensões do espaço de faces para simplificar o entendimento.

Figura 4 - Representação simplificada do espaço de faces



Fonte: Autor

2.6 Métricas de Similaridade

As métricas de similaridade medem o quanto dois objetos são parecidos por meio da distância entre eles [10]. As distâncias entre as representações do espaço de características são utilizadas para tomar as decisões de reconhecimento [20].

2.6.1 Distância Euclidiana

Uma das abordagens mais utilizadas para se calcular a distância entre duas imagens é a distância euclidiana. Dado que A e B são pontos no espaço N dimensional, então pode-se encontrar a distância Euclidiana entre A e B através da Eq.(2.1) [10, 20, 21]:

$$D_E(A, B) = \sqrt{\sum_{i=1}^N (A_i - B_i)^2} \quad (2.1)$$

2.6.2 Distância Manhattan

Distância Manhattan, *City-Block Distance* ou distância retilínea, representa a distância entre pontos como se fosse em formatos horizontais e verticais. Uma analogia seria ruas ou estradas que carros passam, em contrapartida da distância euclidiana que seria apenas uma reta entre os pontos. Essa distância é definida pela Eq. (2.2) [10, 20, 21]

$$Dm_{tt}(A, B) = \sum_{i=1}^N (|A_i - B_i|) \quad (2.2)$$

2.6.3 Distância Chebyshev

Distância Chebyshev ou distância máxima, é uma métrica definida em um espaço vetorial que a distância é escolhida a partir da maior diferença entre dois vetores ou pontos ao longo de todas as coordenadas. A Distância Chebyshev entre dois pontos A e B é [10, 20, 21]:

$$Dc(A, B) = \max(|A_i - B_i|) \quad (2.3)$$

2.6.4 Distância de Mahalanobis

A distância de Mahalanobis é uma métrica que determina a similaridade entre objetos em relação a distribuição dos dados no espaço de entrada. A distância de Mahalanobis pode ser considerada uma distância euclidiana normalizada. Dados dois pontos, A e B, a distância de Mahalanobis é dada pela Eq. 2.4 [9]:

$$D_m(A, B) = \sqrt{\sum_{i=1}^N \frac{(A_i - B_i)^2}{\sigma_i^2}} \quad (2.4)$$

Em que,

- σ_i é o desvio padrão da amostra.

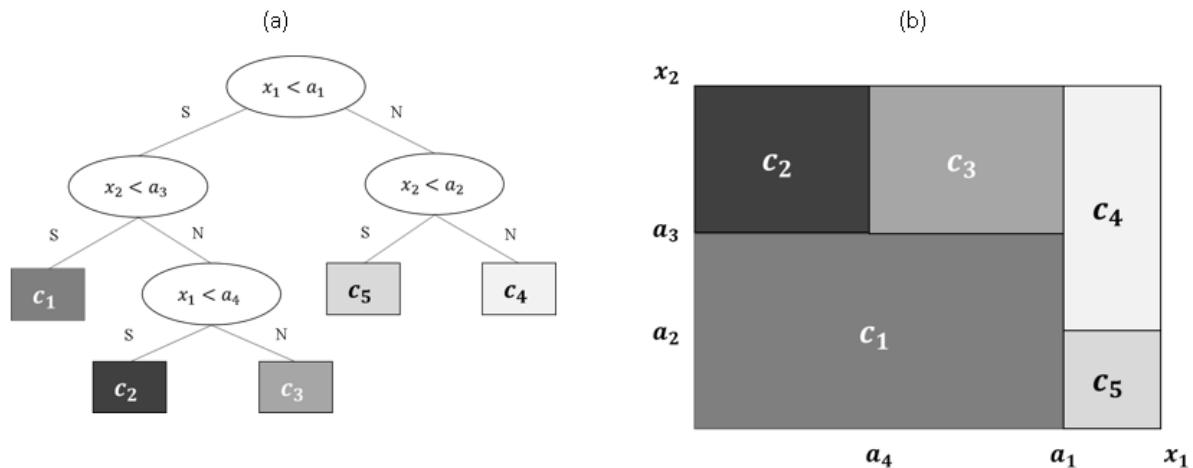
2.7 Árvore de Decisão (DT)

As árvores de decisão, do inglês *Decision Tree* (DT) são algoritmos de aprendizado supervisionado baseados em procura, ou seja, um problema de ML pode ser formulado como um problema de procura em um espaço de possíveis soluções. Podem ser usadas em tarefas de classificação e regressão. Estes modelos utilizam a estratégia de dividir para conquistar, em que um problema complexo é decomposto em subproblemas mais simples e recursivamente esta técnica é aplicada a cada subproblema. No caso das árvores de decisão binária que são utilizadas nesse trabalho, cada nó intermediário divide-se exatamente em dois nós descendentes: o nó

esquerdo e o nó direito. Com cada divisão sucessiva, os membros dos conjuntos resultantes se tornam cada vez mais parecidos entre si, de tal forma, que os subconjuntos vão ficando cada vez mais puros. Um subconjunto dos dados será mais puro na medida em que contém menos classes (ou apenas uma) da variável alvo [2, 22, 23].

DT, de um modo geral, em computação, são estruturas de dados formadas por um conjunto de elementos que armazenam informações chamadas “nós”. Na Figura 5 (a) é apresentada uma arquitetura básica de uma DT, composta pelos nós de divisão (elipse) que contêm um teste condicional baseado no valor do atributo. O nó que não possui filho é conhecido como nó folha (retângulo) ou terminal e representam a decisão a ser tomada. As regiões de separação de classes produzidas por esses classificadores são todas hiper-retângulos, assim como mostrado na Figura 5 (b).

Figura 5- Arquitetura de uma árvore de decisão e as regiões de decisão no espaço de objetos



Fonte: Modificado de [2]

Na elaboração da DT, procura-se relacionar a cada nó de decisão o atributo “mais informativo” entre aqueles ainda não utilizados no caminho desde a raiz da árvore. Entretanto, cada algoritmo tem o seu próprio método para distinguir o atributo mais informativo, fazendo com que a topologia e a qualidade da árvore variem em função do algoritmo utilizado. Todo o algoritmo foi projetado para otimizar cada divisão da DT para maximizar a pureza. Pode-se pensar em pureza como o quão homogeneizados são os agrupamentos. Dependendo de qual medida de impureza é usada, o resultado da classificação em árvore pode variar. Duas métricas comuns, usados para medir a impureza de um nó, são os índices de Gini e a Entropia [18].

A Entropia é o cálculo do ganho de informação baseado em uma medida utilizada na teoria da informação. A Entropia caracteriza a impureza dos dados: em um conjunto de dados,

é uma medida do grau de desordem dos dados de entrada em relação à sua classificação. Por exemplo, a Entropia é máxima quando o conjunto de dados é heterogêneo, ou seja, cada elemento pertence a uma classe diferente. Dado um conjunto de entrada (S) que pode ter c classes distintas, a Entropia de S será dada por:

$$Entropia(S) = \sum_{i=1}^c -p_i \log_2 p_i \quad (2.5)$$

Onde: p_i é a porcentagem de dados em S que pertencem à classe i [24].

O índice Gini, desenvolvido por Conrado Gini em 1912, mede o grau de heterogeneidade dos dados. Logo, pode ser utilizado para medir a impureza de um nó. Este índice num determinado nó é dado por:

$$Índice_{Gini} = 1 - \sum_{i=1}^c p_i^2 \quad (2.6)$$

Em que p_i é a probabilidade de cada classe em cada nó e c é o número de classes [24].

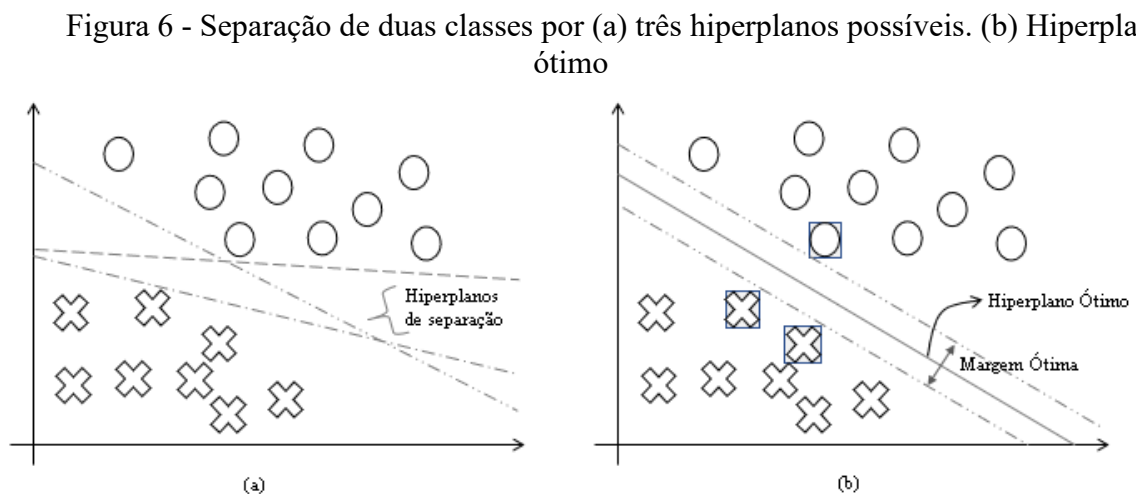
Uma vez construída uma DT, seu uso é imediato e muito rápido computacionalmente. Outra vantagem é que o aumento na quantidade de dimensões não afeta seu desempenho. Além disso, a facilidade de interpretação e visualização de uma árvore de decisão é um de seus pontos fortes. No entanto, a construção de uma árvore de decisão, chamado de processo de indução, pode ser uma tarefa de alta demanda computacional [25]. Outra desvantagem desse modelo é o sobreajuste (*overfitting*), que consiste na adaptação extrema do modelo aos dados de treinamento [24]. Quando não é definido nenhum limite para a árvore de decisão, ela pode, no pior caso, criar uma folha para cada amostra de treinamento, tornando-se específica para o conjunto de dados utilizado. Para prevenir o sobreajuste, pode-se impor restrições na construção da árvore, como o número mínimo de amostras para uma divisão de nós, o número mínimo de amostras para um nó folha ou a profundidade máxima.

2.8 Máquina de Vetores de Suporte (SVM)

A máquina de vetores de suporte, do inglês *Support Vector Machine* (SVM) constitui uma técnica embasada na Teoria de Aprendizado Estatístico [26]. É um algoritmo de classificação supervisionado que, por meio de vetores de amostras de treinamento, estabelece um hiperplano ótimo de separação entre as classes a fim de maximizar a distância entre elas [27].

Um hiperplano ótimo é definido como a função de decisão com margem máxima entre os vetores das duas classes. A margem máxima é obtida encontrando o hiperplano que possui a maior distância até o ponto de dados de treinamento mais próximo de qualquer classe. Na Figura 6, são mostradas duas classes, círculos e cruzes, a qual mostra a ideia básica de um SVM linear. Na Figura 6 (a) são mostradas linhas tracejadas, as quais representam alguns possíveis hiperplanos que podem ser utilizados para separar as duas classes, porém estes não apresentam uma boa margem. Já na Figura 6 (b) é mostrado um hiperplano ótimo, ou seja, apresenta a maior margem. Desta forma pode haver infinitos hiperplanos de separação entre classes, mas apenas um possui a melhor margem de separação.

Obter um hiperplano que possua uma grande margem de separação significa ter um menor erro de generalização do classificador [28]. Para construir esses hiperplanos, é necessário considerar uma pequena quantidade dos dados de treinamento, os chamados vetores de suporte, que são usados para a determinação dessa margem. No caso da Figura 6 são representados pelas formas contornadas por quadrados.



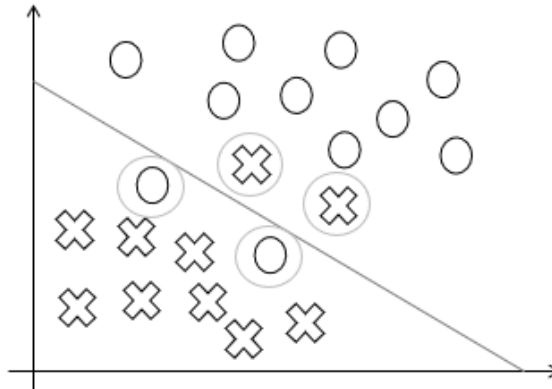
Fonte: Autor

2.8.1 SVM de margem suave

Em conjunto de dados obtidos em problemas reais em sua grande maioria as classes não são linearmente separáveis, isso devido ao ruído inerente e *outliers* (atributos que pertencem a uma classe que se encontram distantes dos demais). Desta forma fica inviável a obtenção de um hiperplano que separe duas classes. Como alternativa são adicionadas variáveis de folga que representam o deslocamento que os padrões classificados erroneamente encontram-se do limite da margem de separação referente a sua classe. Essas variáveis permitem quantificar o número de padrões classificados erroneamente. Uma penalidade é associada a estes erros com o uso do

parâmetro de regularização C . [29]. A Figura 7 ilustra um exemplo de um conjunto não linearmente separável onde é inserida uma folga na margem permitindo o treinamento com uma parcela de erros.

Figura 7- Representação do hiperplano de separação para padrões não linearmente separáveis

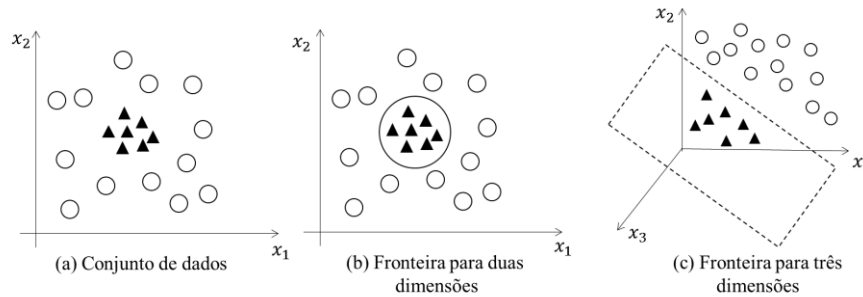


Fonte: Autor

2.8.2 SVM não-linear

As SVMs geralmente lidam com classificação de padrões lineares ou não-lineares. Padrões lineares são padrões que são facilmente distinguíveis ou podem ser separados, em sua dimensão original, por uma superfície linear. Para separar padrões não-lineares, realiza-se um mapeamento do conjunto de treinamento, movendo o espaço original para um novo espaço de dimensão superior. A Figura 8 mostra um exemplo deste processo de mapeamento, onde em (a) tem-se o conjunto de treinamento não-linear em um espaço bidimensional; em (b) é mostrado a fronteira não-linear entre as classes em seu espaço original; em (c), o espaço original foi transformado em um conjunto linearmente separável o qual pertence a um espaço tridimensional. Para a realização desse mapeamento dos dados de entrada para o novo espaço de maior dimensão utiliza-se uma função *kernel* que satisfaça a condição de Mercer, detalhado em [30, 31]. Algumas funções *kernel* frequentemente usadas na solução de problemas gerais são funções de base radial (*Radial Basis Function* - RBF), sigmoidais e polinomiais. Cada um desses *kernels* possui parâmetros que devem ser especificados pelo usuário. No Quadro 1 é possível visualizar algumas destas funções juntamente com seus respectivos parâmetros. Onde $d \in N^*$ e $\gamma \in R_+^*$ são parâmetros nos kernels RBF, Sigmoidal e Polinomial. Neste trabalho os parâmetros d e γ são representados por *Degree* e *Gamma* respectivamente.

Figura 8 - Exemplo de transformação realizada em um conjunto de dados não linear para um novo espaço de características de maior dimensão



Fonte: Modificada de [2]

Quadro 1 - Exemplo de funções de *kernel* e seus respectivos parâmetros

Tipo de kernel	Função $K\langle x_i, x_j \rangle$	Parâmetros
RBF	$\exp\left(-\gamma\ x_i - x_j\ ^2\right)$	γ
Sigmoidal	$\tanh(\gamma\langle x_i, x_j \rangle)$	γ
Polinomial	$(\gamma\langle x_i, x_j \rangle)^d$	γ, d

Fonte: Modificado de [2]

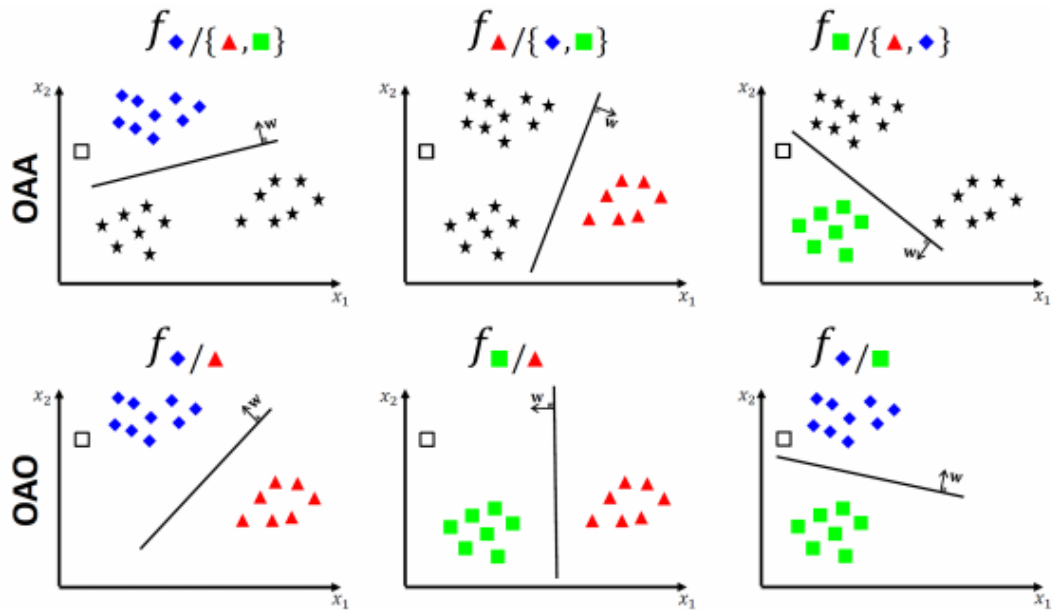
2.8.3 SVM multiclasse

O SVM em si foi desenvolvido para classificar problemas binários, ou seja, com duas classes. Existem várias técnicas que possibilitam a classificação de n classes [31]. Serão abordadas duas técnicas, sendo elas “Um-Contra-Todos” (*One-Against-All* - OAA) e “Um-Contra-Um” (*One-Against-One* - OAO) que são baseadas em decomposições binárias.

Na Figura 9 temos um problema de classificação envolvendo três classes (losangos, triângulos e quadrados preenchidos), o quadrado sem preenchimento representa um ponto de dados sendo classificado. A primeira linha de gráficos se utiliza da técnica OAA para a classificação e a segunda utiliza-se da técnica OAO. A técnica OAA consiste em definir uma quantidade de hiperplanos relativos à quantidade de classes existentes, cada hiperplano é definido pelo treinamento de uma classe específica com as demais (representados pelas estrelas na Figura 9). Para a classificação de um ponto de dados é dada através da obtenção do SVM que apresenta maior valor de sua função de decisão [29]. No exemplo da Figura 9, o ponto de dados seria classificado como losango azul, pois se encontra mais distante do respectivo hiperplano.

Já na técnica OAO, para o mesmo problema de classificação, o número de hiperplanos definidos é igual à combinação das classes dois a dois. Cada classificador é responsável pela separação dos padrões entre um determinado par de classes. A classificação de um ponto de dados pode ser obtida em uma mesma classe diversas vezes. Neste caso, a classificação de um padrão baseia-se na quantidade de vezes que o mesmo é associado a uma determinada classe [29]. No exemplo da Figura 9 o ponto de dado seria classificado como losango azul.

Figura 9 - Exemplo de aplicação de técnicas multiclasse OAA e OAO em um problema de separação envolvendo 3 classes



Fonte: [29]

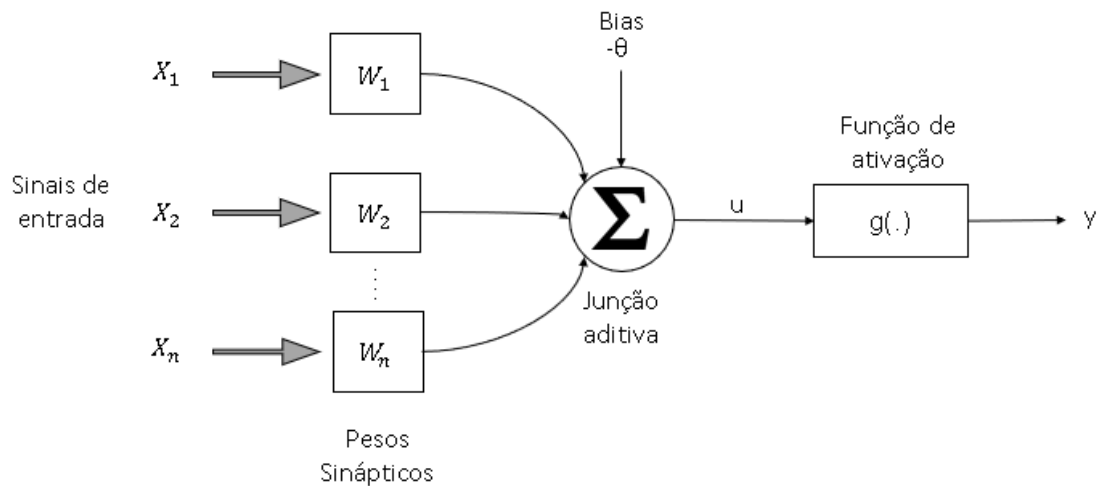
2.9 Redes Neurais Artificiais

As Redes Neurais Artificiais (RNA) são sistemas de aprendizado inspirados no comportamento do cérebro humano. As principais propriedades estão relacionadas à capacidade de aprender por exemplos, de interpolar com base em padrões fornecidos e de selecionar propriedades específicas dentro do universo amostral. Uma RNA também pode ser definida como um sistema massivamente paralelo e distribuído, composto por unidades de processamento simples que possuem uma capacidade natural de armazenar e utilizar conhecimento [27].

Um neurônio artificial é uma unidade de processamento de informação que é fundamental para operação de uma rede neural [27]. A Figura 10 mostra um modelo em diagrama de blocos de um neurônio artificial que é a base das redes neurais artificiais. O neurônio é descrito por um conjunto de X_n entradas, as quais são multiplicadas por um

determinado peso W_n , estes chamados de pesos sinápticos e, em seguida, os resultados são somados e comparados a um limiar ou função de ativação [32].

Figura 10 - Diagrama de blocos de um neurônio artificial



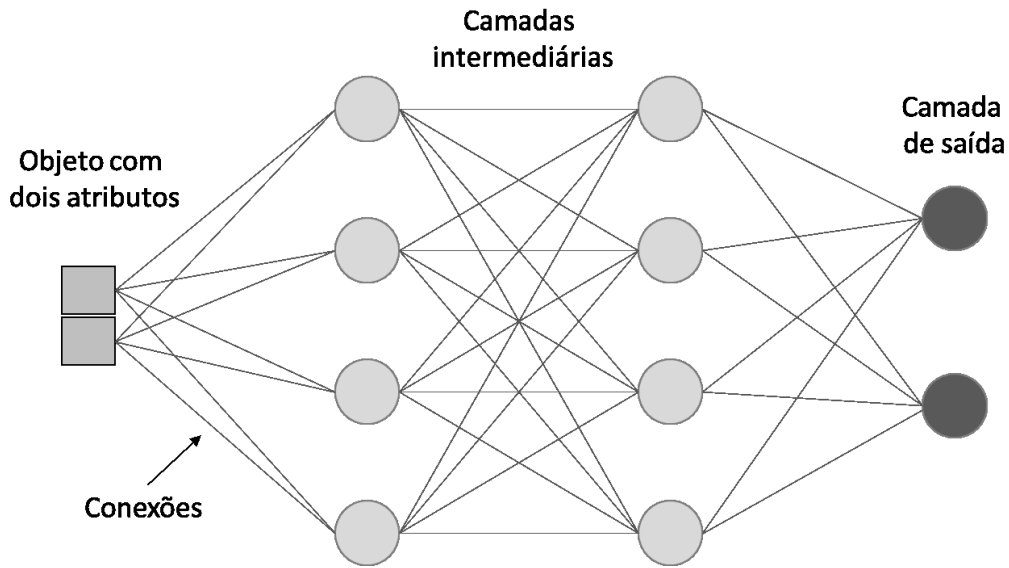
Fonte: Autor

O aprendizado de uma RNA é realizado por meio de processos iterativos de ajustes aplicados aos pesos sinápticos, o chamado treinamento. O aprendizado só ocorre quando a rede neural atinge uma solução generalizada para um determinado problema [33]. A RNA possui dois tipos de sinapses: inibitória ou excitatória. A sinapse inibitória ocorre quando o valor da função de ativação fica abaixo de um dado limiar, resultando em uma resposta negativa. A sinapse excitatória ocorre quando o valor da função de ativação fica acima de um dado limiar, resultando na ativação do neurônio ou resposta positiva.

O Perceptron foi o primeiro modelo de uma rede neural artificial que demonstrou matematicamente a possibilidade de simulação do aprendizado cerebral. Este modelo simulava um neurônio com entradas e pesos que, ajustados teriam o "poder" de aprender a se comportar de determinada forma. Entretanto, este modelo ainda era simplista e era preciso que se conhecessem os pesos para se encontrar determinada saída [34].

Uma limitação das redes de uma camada, como as redes perceptron, é que elas classificam apenas objetos que são linearmente separáveis. Para problemas que não sejam linearmente separáveis uma alternativa é adicionar uma ou mais camadas intermediárias [2]. Uma rede com neurônios interligados por mais de uma camada é denominada de rede multicamada (MLP – *Multilayer Perceptron*). Um algoritmo muito conhecido e responsável pelo aprendizado da rede é denominado de *Backpropagation*, e descrito detalhadamente em [18]. Na Figura 11 é mostrado um modelo de uma rede MLP.

Figura 11 - Exemplo de RNA multicamada típica



Fonte: Modificada de [2]

2.10 Avaliação do Algoritmo

O objetivo do modelo de ML é reconhecer padrões que oferecem um bom resultado de generalização nos dados não vistos, em vez de apenas memorizar os dados mostrados durante o treinamento. Após desenvolver um modelo de aprendizagem, é importante verificar se ele apresenta um bom desempenho em amostras não usadas no treinamento do modelo. Várias métricas são usadas em ML para medir a precisão da predição de um modelo. A escolha da métrica depende da tarefa de ML. É importante analisar essas métricas para decidir se o modelo está com um bom desempenho [35].

2.10.1 Métricas de Classificação

Uma métrica de desempenho muito utilizada na avaliação de um classificador f é a sua taxa de erro ou de classificações incorretas, mostrado na Eq. 2.5, em que $I(a) = 1$ se a é verdadeiro e 0 caso contrário. Dado um conjunto de dados contendo n objetos, sobre o qual a avaliação será realizada, essa taxa equivale à proporção de exemplos desse conjunto classificados incorretamente por f e é obtida pela comparação da classe conhecida de x_i , denotada por y_i com a classe predita $f(x_i)$. Esse tipo de medida equivale ao uso da função 0-1 relacionando os rótulos dos objetos às predições obtidas [2].

$$err(f) = \frac{1}{n} \sum_{i=1}^n I(y_i \neq f(x_i)) \quad (2.7)$$

A taxa de erro varia entre 0 e 1, e valores próximos ao extremo 0 significa que o classificador possui um baixo erro. O complemento dessa taxa correspondente à taxa de acerto ou acurácia do classificador, nesse caso, valores próximos de 1 são considerados melhores [2].

$$a_c(f) = 1 - e_{rr}(f) = \frac{1}{n} \sum_{i=1}^n I(y_i = f(x_i)) \quad (2.8)$$

2.10.2 Matriz de Confusão

A matriz de confusão é uma tabela que permite a visualização do desempenho de um algoritmo. Cada coluna da matriz representa as instâncias em uma classe prevista, enquanto cada linha representa as instâncias em uma classe real (ou vice-versa). O nome deriva do fato de que é fácil ver se o sistema está confundindo duas classes (ou seja, rotulando erroneamente uma como outra).

Logo, cada elemento m_{ij} de uma matriz de confusão M_c apresenta o número de exemplos da classe i classificados como pertencentes a classe j . Para k classes, M_c tem então dimensão $k \times k$. A diagonal principal apresenta os acertos do classificador, enquanto os outros elementos correspondem aos erros cometidos nas suas predições. Por meio da avaliação dessa matriz, pode-se verificar quantitativamente quais classes apresentam maior taxa de erro de classificação [2].

Por exemplo a Figura 12 ilustra uma matriz de confusão em um sistema de classificação de 3 pessoas em que cada pessoa possui 15 amostras. Como é possível observar, a “Pessoa 1” teve todas amostras preditas corretamente, já a “Pessoa 2” foi predita erroneamente 3 vezes como “Pessoa 1” e 2 como “Pessoa 3”. A “Pessoa 3” foi predita erroneamente 2 vezes como “Pessoa 1” e apenas uma vez como “Pessoa 2”.

Figura 12 - Exemplo de uma matriz de confusão para um problema de 3 classes

		Classe predita		
		PESSOA 1	PESSOA 2	PESSOA 3
Classe verdadeira	PESSOA 1	15	0	0
	PESSOA 2	3	10	2
	PESSOA 3	2	1	12

Fonte: Autor

2.10.3 Validação Cruzada

O objetivo da validação cruzada (*Cross-validation*) é testar a capacidade do modelo de prever novos dados que não foram usados na estimativa, a fim de sinalizar problemas como *overfitting* ou viés de seleção e dar uma ideia de como o modelo será generalizado para um conjunto de dados independente (ou seja, um conjunto de dados desconhecido, por exemplo, de um problema real) [36] .

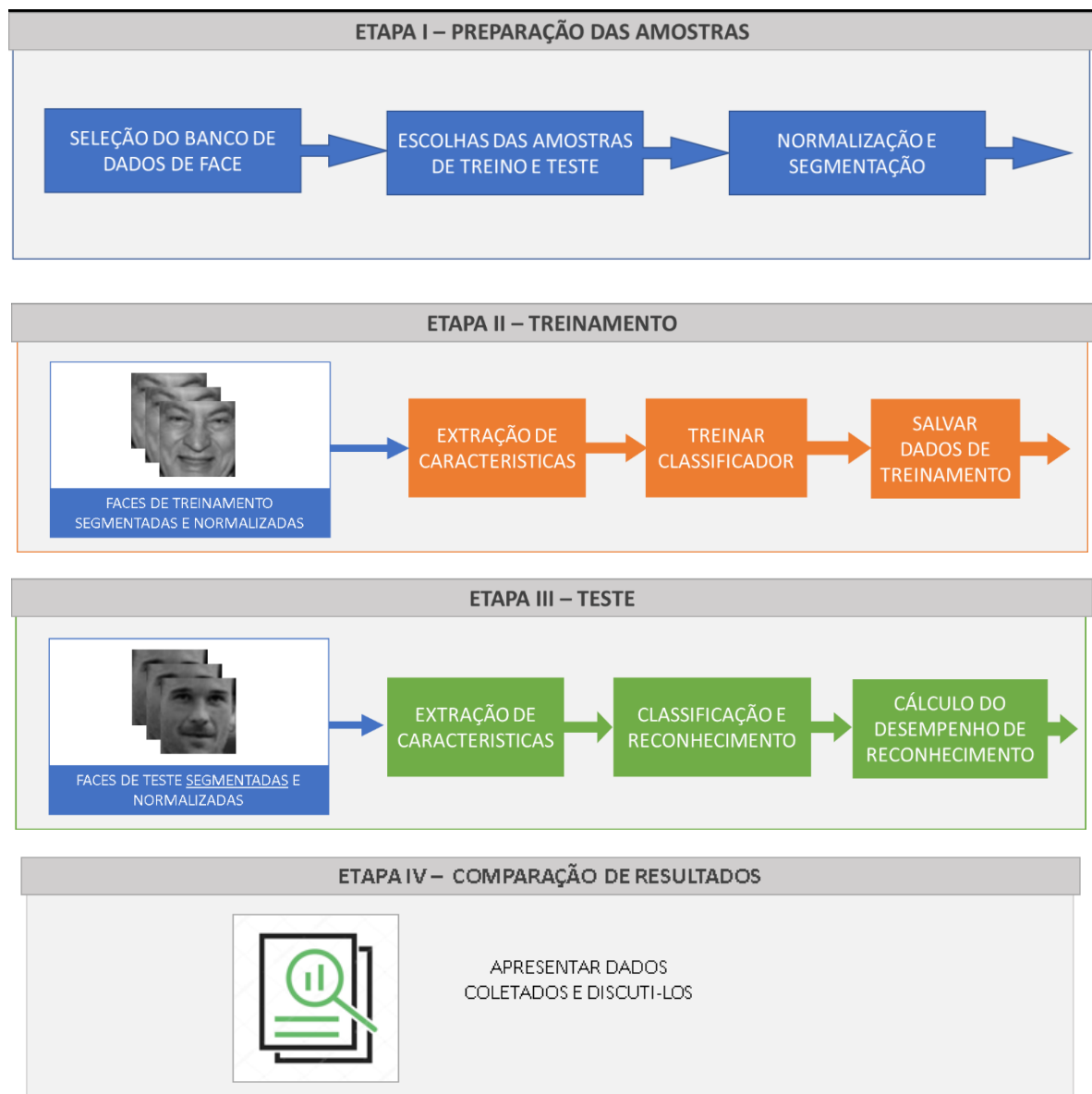
O *k-fold* é o método de validação cruzada mais conhecido e utilizado. Consiste em dividir o seu conjunto de dados normalmente entre 5 a 10 subgrupos contendo uma quantidade de amostras aproximadamente igual. Em seguida separa-se o primeiro dos grupos para ser o grupo de validação ou teste. O restante dos grupos é utilizado para treinar o modelo e aplicar no grupo de teste para comparar o resultado e ver qual foi a taxa de erro. Esse método se repete até que todos os grupos tenham sido utilizados como grupo de teste e todos tenham um percentual de erro. O desempenho final do preditor é dado pela média dos desempenhos observados sobre cada conjunto de teste [2].

Capítulo 3

Metodologia

O desenvolvimento deste trabalho consiste na realização de quatro etapas como mostrado na Figura 13. O processo usado é o de aprendizado supervisionado, onde nas estruturas das imagens dos bancos de dados temos um identificador para cada indivíduo, seguido das várias expressões feitas por cada um deles, sendo possível medir o nível de acerto do algoritmo com relação às faces usadas para teste e também saber quais faces foram detectadas erroneamente. A seguir será descrito como foi realizada cada etapa separadamente.

Figura 13 - Etapas do desenvolvimento deste trabalho



3.1 ETAPA I: Preparação das amostras

O banco de dados de faces escolhido para a maioria dos experimentos foi o Yale Face Database [37], constituído de 165 imagens faciais de 15 indivíduos distintos sendo que 14 são homens e uma mulher. Cada indivíduo possui 11 amostras de imagens faciais com variações de expressões e iluminação. Os motivos que resultaram na escolha deste banco dados foram:

- Possuir muitas amostras por classe e de forma uniforme, ou seja, possuir a mesma quantidade de amostras por classe.
- Possuir apenas variações de expressões e iluminação. Com menos variações é possível avaliar com mais precisão os algoritmos, dado o objetivo do trabalho.

A escolha das amostras de treino e teste é realizada por meio a implementação do método de amostragem por validação cruzada em que se utiliza da técnica *k-fold*. O valor de *k* foi definido como igual a onze, isso para que seja possível utilizar o máximo de imagens no treinamento, desta forma, cada modelo é treinado com 10 imagens de cada classe e testado com uma imagem.

Na fase de normalização e segmentação foi realizado o recorte e redimensionamento das faces para que o fundo das imagens não entre como atributo das classes, e que todas imagens faciais possuam mesmo tamanho, assim como mostrado na Figura 14. Em seguida, foi feita a equalização de histograma para que variações de contraste não influenciem no processo de FR.

Figura 14 - Segmentação de face



Fonte: Autor

3.2 ETAPA II: Treinamento

Esta etapa se resume em realizar o treinamento do algoritmo com as imagens de treino selecionadas e normalizadas na etapa anterior. O método de extração de características utilizado para treinamento e FR em todo o projeto foi a técnica PCA para reduzir a dimensão eliminando a redundância de informação nas imagens. Cada modelo de aprendizagem gerado foi obtido da combinação do vetor de características obtido com o cálculo do PCA e os seguintes procedimentos de treinamento:

- Treinamento com DT:

Os modelos de treinamento foram realizados utilizando os índices Gini e Entropia. Cada critério foi treinado com a variação da máxima profundidade e do número mínimo de amostras necessárias para estar em um nó folha.

- Treinamento com SVM:

Os modelos de treinamento foram realizados utilizando 4 funções de *kernel*: Linear, Polinomial, RBF e Sigmoides. Diferentemente do *kernel* Linear, cada um desses *kernels* citados possui parâmetros a serem definidos. Vários modelos foram obtidos com a variação desses parâmetros. Outro parâmetro que foi variado é a constante de rigidez, a qual está relacionada com a suavidade das margens.

- Treinamento com Rede Neural Artificial:

A estrutura da rede neural artificial foi MLP em que a quantidade de neurônios e camadas foram variados. O algoritmo de treinamento da rede foi o *Backpropagation*. Os modelos de treinamento foram realizados utilizando 4 funções de ativação: Identidade, Sigmoides, Tangente Hiperbólica e Unidade Linear Retificada (ReLU).

3.3 ETAPA III: Teste

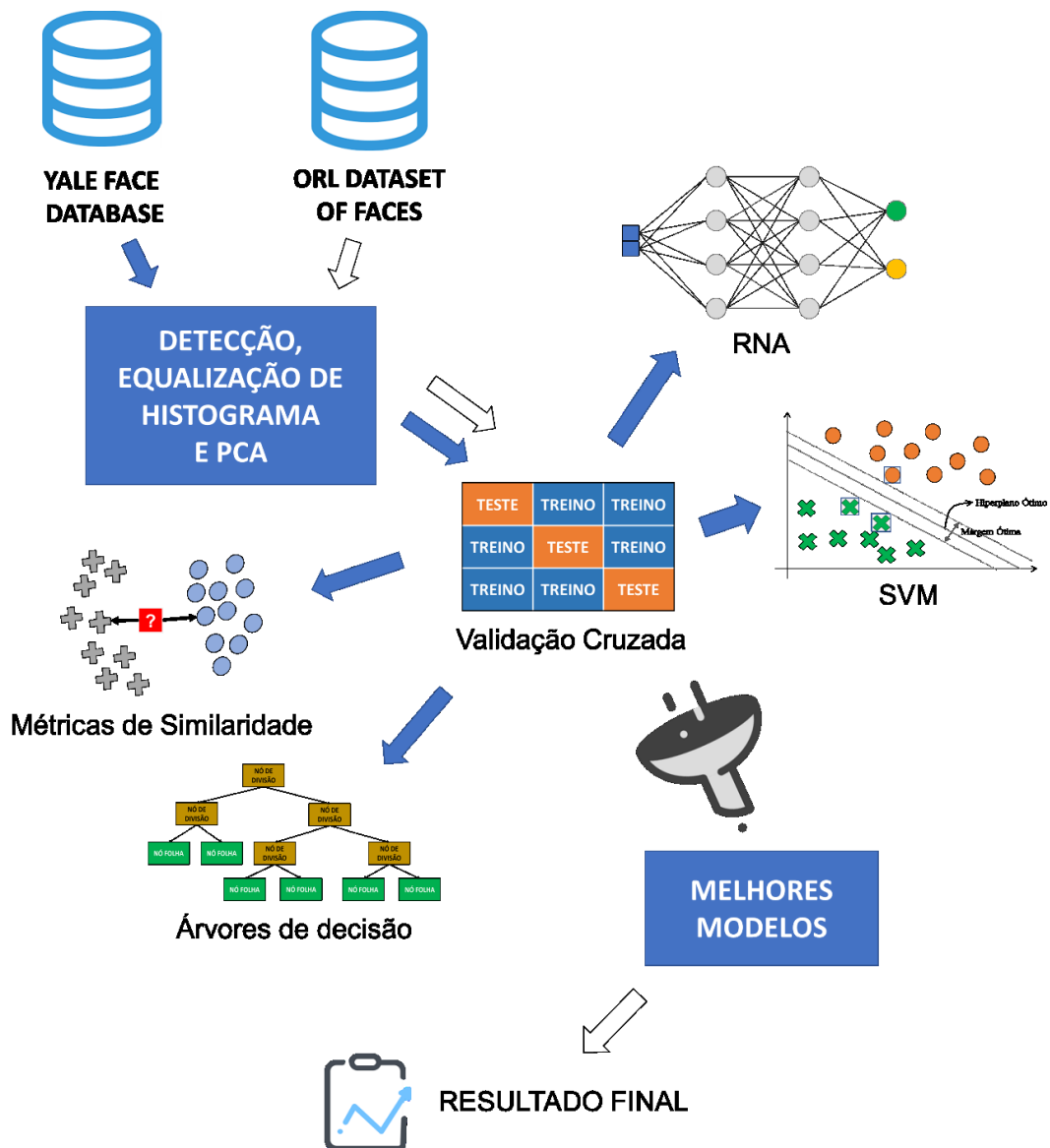
Realização de testes dos algoritmos de classificação. A partir das imagens de testes escolhidas e normalizadas na etapa I, foi realizada a extração do vetor de características com o método PCA. Em seguida foi realizada a predição/classificação das amostras de teste de cada modelo treinado na etapa anterior e também utilizando os classificadores de distância: Euclidiana, Manhattan, Chebyshev e Mahalanobis. Por fim foi calculada a acurácia média, desvio padrão das acurácias e tempo de processamento da validação cruzada de cada um dos modelos obtidos.

3.4 ETAPA IV: Comparação de Resultados

Na Figura 15 é mostrado de uma forma geral, como é organizado o fluxo de testes para a análise comparativa dos modelos de classificadores. Inicialmente as amostras de face do banco Yale Face Database passa pelos processos de Detecção Facial, Equalização de Histograma e suas características principais são obtidas com a técnica PCA. As acurácias dos modelos com classificadores utilizando métricas de similaridade, DT, SVM e RNA são calculadas para cada *fold* de teste do processo de validação cruzada. Com as acurácias obtidas,

seleciona-se o melhor modelo de cada classificador utilizado. O melhor modelo é escolhido pelo que demonstra maior taxa de acurácia, caso de empate escolhe-se o que demonstrar menor desvio padrão. Caso o empate persista, o critério de desempate é dado pelo modelo que registra o menor tempo para o processamento de todos treinos e testes da validação cruzada. Com os melhores modelos obtidos é processada uma matriz de confusão para a visualização do desempenho dos algoritmos por amostra. Em seguida todo o procedimento até aqui explicado é repetido com o banco de dados *ORL Dataset Of Faces*, porém apenas com as melhores configurações de classificadores obtidos com o banco *Yale Face Database*. Assim é comparado quais dos melhores modelos selecionados apresentam melhores resultados sem que seja necessária a reconfiguração dos parâmetros dos mesmos.

Figura 15 – Fluxo dos testes realizados para a comparação dos classificadores



Capítulo 4

Recursos

4.1 Linguagem Python

Escolheu-se a linguagem Python para a implementação deste trabalho devido a sua popularidade, por possuir muitas comunidades ativas, simplicidade, possuir inúmeras bibliotecas disponíveis que podem contribuir na velocidade de implementação e coleta de resultados [38].

4.2 Bibliotecas

- *Scikit Learn 0.21*

A biblioteca *Scikit Learn* fornece implementações de última geração de muitos algoritmos de ML bem conhecidos, mantendo uma interface fácil de usar totalmente integrada com a linguagem Python [39].

- *OpenCV 4.1.1*

O OpenCV (*Open Source Computer Vision Library*) é uma biblioteca de software de visão computacional e ML em código aberto. O OpenCV é construído para fornecer uma infra-estrutura comum para aplicativos de visão computacional. Sendo um produto licenciado pela Berkeley Source Distribution (BSD). o OpenCV torna fácil para os interessados utilizarem e modificar o código [40].

4.3 Máquina utilizada para realização dos testes

- Notebook – Modelo: Acer Aspire E1-572-6_BR648;
- Sistema Operacional: Windows 10 Pro 64bits;
- Processador: Intel Core i5-4200U;
- Memória RAM: 6Gb DDR 3 L;
- HD: 240 GB SSD Kingston.

4.4 Banco de dados de imagens faciais

Os bancos de dados de face são utilizados em sistemas de testes para análise da taxa de reconhecimento de um algoritmo de FR [41]. Junto com o desenvolvimento de algoritmos de FR, um número relativamente grande de bancos de dados foi criado. Muitos desses bancos de

dados que são desenvolvidos, possuem características distintas os quais podem ser adaptados às necessidades específicas do algoritmo em desenvolvimento [12].

No Quadro 2 são indicadas algumas bases de dados encontradas em seus respectivos repositórios. A coluna “nome” identifica o banco de faces, a coluna “total de imagens” indica a quantidade de imagens de faces possui o banco, “Indivíduos” indica a quantidade de pessoas presentes nas imagens do banco e também a proporção de homens e mulheres, “Amostras” indica a quantidade de imagens por pessoa [10]. Já a coluna “Condições” especifica as variações de aparência contidas no banco de faces, e estas estão divididas nas seguintes condições:

Ambiente: As imagens contêm variações do plano de fundo.

Expressões: As imagens contêm diversas expressões faciais como alegria, tristeza, raiva entre outras.

Iluminação: As imagens contêm variações na distribuição de iluminação do ambiente.

Obstruções: As imagens contêm diversos tipos de obstruções faciais como óculos, chapéu, etc.

Poses: As imagens contêm variações na inclinação da face do indivíduo.

Tempo: As imagens contêm variações de tempo de um mesmo indivíduo podendo apresentar características de envelhecimento.

Quadro 2 - Bancos de imagens faciais

Nome	Total de Imagens	Individuos	Amostras	Condições					
				Ambiente	Expressões	Iluminação	Obstruções	Poses	Tempo
CVL Database	798	114 (108/6)	7		X			X	
FacePix Database (ASU)	16290	30	181			X		X	
FEI Face Database	2800	200 (100/100)	14		X	X		X	
Indian Face Database	440	61 (39/22)	~11		X			X	
Labeled Faces in Wild (LFW)	13233	13233	≥1	X	X	X	X	X	X
ORL Database of faces	400	40 (36/4)	10		X	X		X	X
Psychological Image Collection at Stirling (PICS)	312	35	9			X		X	
The AR Face Database	4000	126 (70/56)	26		X	X	X		X
The Facial Recognition Technology (FERET)	14126	1199	~11	X	X	X		X	X
The Japanese Female Facial Expression (JAFFE)	213	10 (0/10)	7		X	X		X	
The Sheffield Face Database	564	20 (16/4)	26-54					X	
Yale Face Database	165	15 (14/1)	11		X	X			
Yale Face Database B	5670	10	576			X		X	
UND Biometrics	33287	487	68	X	X	X		X	X

Modificado de [10]

Capítulo 5

Experimentos e Resultados

Neste capítulo são detalhados todos os procedimentos seguidos para a preparação e realização dos experimentos. Os resultados obtidos dos experimentos estão organizados pelo método utilizado e apresentados em formas de tabelas e gráficos, e após os mesmos, são realizadas análises e interpretações.

5.1 Segmentação e Equalização de Histograma

Esta etapa consiste em normalizar e selecionar a região de interesse das amostras de face do banco de imagens utilizado, organizando as amostras de face para que estejam no formato ideal a serem utilizados como parâmetros nas funções de classificação da biblioteca *scikit-learn*.

Realizando uma investigação inicial sobre as imagens do banco Yale Datafaces pode-se observar que todas elas possuem dimensões padronizadas, cada imagem sendo uma matriz de tamanho 320×243 *pixels* em escala de cinza de 256 níveis. Os nomes dos arquivos de imagem seguem um padrão que possui um identificador da pessoa seguidamente da expressão ou disposição da iluminação contida na imagem. Por exemplo, o arquivo *subject01-surprised* contém uma imagem da pessoa 1 com expressão de surpresa.

Para recortar as imagens de face apenas na região de interesse foi utilizado o classificador em cascata de Haar citado na Seção 2.2. Para isso utilizou-se a função *cv2.CascadeClassifier* da biblioteca OpenCV. É necessário carregar na função um arquivo no formato *.xml* que contém as características de Haar correspondente a um classificador treinado para detecção facial, disponibilizado em [42]. As imagens recortadas com esse método não possuem dimensões iguais, assim, para manter as imagens com a mesma quantidade de atributos (*pixels*) redimensionou-se todas as imagens obtidas utilizando interpolação linear para um tamanho padrão de 150×150 *pixels* que corresponde a uma quantidade total de 22.500 *pixels*. Este tamanho foi escolhido devido as imagens recortadas apresentarem tamanhos relativamente próximos. A interpolação foi realizada por meio da função *cv2.resize* da biblioteca OpenCV.

Após a selecionar a região de interesse das amostras de face realizou-se a equalização de histograma das mesmas para reduzir o efeito da variação de iluminação no ambiente. Para isso foi utilizada a função *cv2.equalizeHist* da biblioteca OpenCV. A Figura 16 mostra a comparação das imagens do banco de faces antes e após o recorte e equalização do histograma.

Figura 16 - Comparação de imagens do banco de faces (a) antes e (b) após o recorte e equalização do histograma.



Fonte: Autor

5.2 Redução de dimensionalidade com PCA

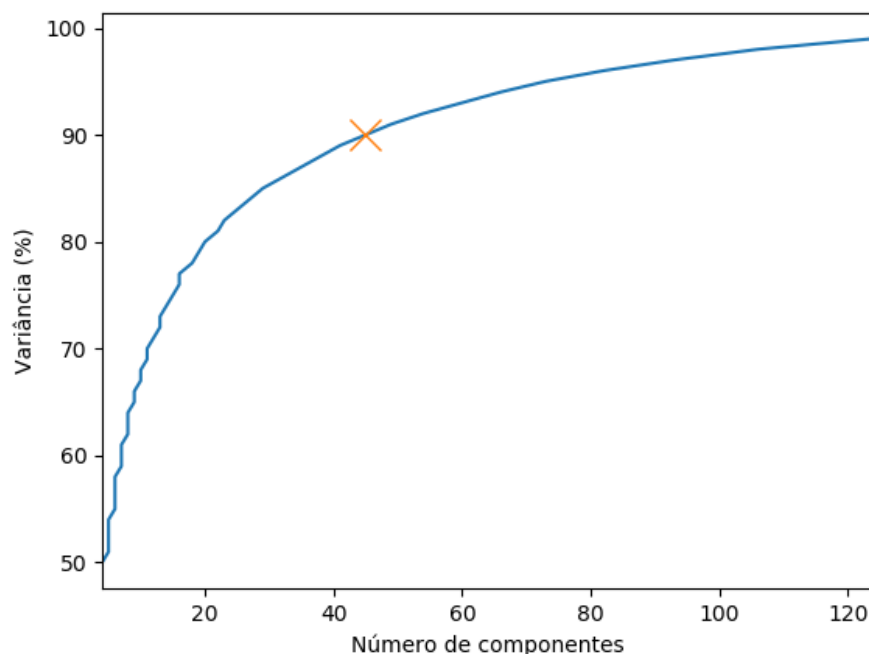
Nesta fase temos 165 imagens de faces, cada uma com 22.500 *pixels* (vetor de características). Com o objetivo de reduzir a dimensão das amostras, é possível gerar descritores das faces. A face humana indica um padrão de características comuns presentes na maioria das pessoas, tais como olhos, boca e nariz. Para a abordagem holística, essas características presentes nas imagens de face são as variações de aparência causadas por mudanças de iluminação, expressões faciais e poses.

A técnica PCA representa essa variabilidade da aparência em um conjunto de componentes. Com PCA o vetor de intensidade de pixels de cada imagem pode ser convertido em um vetor, em que, cada dimensão contém uma percentagem da variância da imagem original. Sendo que a primeira componente deste vetor possui maior percentagem de variância. Essa percentagem é distribuída de forma decrescente entre as demais componentes. Assim o primeiro passo deste experimento é determinar a quantidade de componentes a serem utilizadas para representar cada amostra [10].

Neste experimento recorreu-se à classe *PCA* contida no pacote *sklearn.decomposition* da biblioteca *scikit-learn* para realizar todos cálculos envolvendo PCA. Esta classe permite definir a quantidade de componentes de forma manual ou informando como parâmetro o

percentual de variância. Assim o método calcula automaticamente o número de componentes necessários para corresponder à variância inserida. Para definir o número de componentes, primeiramente traçou-se o gráfico relacionando o percentual de variância com o número de componentes necessários para atingir a mesma. O gráfico pode ser observado na Figura 17. Esse gráfico possui um ponto em destaque indicando que com 45 componentes principais é possível explicar 90% da variabilidade das amostras. Esta quantidade de componentes foi escolhida para verificar se é possível obter bons resultados utilizando apenas 0,2% do total de informação extraída da imagem facial. O motivo de não usar 100% da variância é porque corresponde a todos os 22.500 componentes [43].

Figura 17 - Relação do número de componentes com a variância da base de dados Yale Face Database



Fonte: Autor

A redução da dimensionalidade dos dados é obtida utilizando as funções *PCA.fit* e *PCA.transform*. Antes e após a transformação dos dados realizou-se a normalização das imagens de face para que cada amostra da base tenha média igual a “0” e desvio padrão igual a “1”. Esse procedimento de normalização da escala foi realizado por meio da classe *StandardScaler* do pacote *sklearn.preprocessing*. Este procedimento é muito importante quando os atributos do vetor de características possuem intervalos distintos eliminando assim qualquer viés que um atributo específico pode conter. Para o caso das amostras de face utilizadas não seria necessário pois todos atributos possuem valores dentro do mesmo intervalo

de $[0, 255]$. No entanto, alguns dos classificadores a serem utilizados são sensíveis à escala dos atributos, tais como SVM e RNA [44, 45], desta forma, a normalização é recomendada.

5.3 Validação cruzada utilizando a biblioteca *scikit-learn*

No pacote *sklearn.model_selection* da biblioteca *scikit-learn* existe alguns métodos para processar a validação cruzada. Cada método retorna informações distintas sobre o modelo a ser testado. Para a maioria dos experimentos utilizou-se o método *cross_val_score* que retorna um vetor contendo a acurácia (caso não especifique outra métrica) obtida em cada teste realizado na validação. Nesse método é necessário informar um objeto do tipo *estimator*, o qual é utilizado especificar as características classificador. Também é necessário informar em um parâmetro x um vetor ou lista contendo o conjunto de dados do problema, e um parâmetro y onde especifica a qual classe cada objeto do conjunto de dados x pertence. Outro parâmetro importante e denominado *cv* onde se especifica a quantidade de partições (*folds*) para a validação, que em todos experimentos definiu-se igual a 11. Após o cálculo da validação cruzada são calculados a média e o desvio padrão das acurácias. Também se mediu o tempo da realização dos cálculos da validação cruzada. A validação cruzada foi utilizada em todos os experimentos deste trabalho.

5.4 Experimentos utilizando métricas de similaridade

Para a realização deste experimento fez-se necessário o uso do classificador *k-nearest neighbors* (*k*-NN). O *k*-NN calcula a distância do objeto de teste para todos os elementos do conjunto de treino. Em seguida, separa os k objetos de treino mais próximos do objeto de teste. Por fim, a saída pode ser calculada como a classe com a frequência mais alta dos k objetos mais próximos, (k é um número inteiro positivo, geralmente pequeno). Se $k=1$, o objeto de teste é simplesmente atribuído à classe do único vizinho mais próximo [46].

A aplicação do método com a biblioteca *scikit-learn* é realizada por meio da classe *KNeighborsClassifier* contida no pacote *sklearn.neighbors*. O valor de k e a métrica de similaridade utilizada podem ser definidos nos atributos *n_neighbors* e *metric* respectivamente. O valor de k foi definido como igual a 1 e as métricas testadas foram as seguintes: “Euclidiana”, “Manhatan”, “Chebyshev” e “Mahalanobis”. Em cada teste utilizou validação cruzada, a implementação é explicada na Seção 5.3. No Quadro 3 é mostrado a acurácia, desvio padrão e tempo de processamento obtidos, por meio do processo da validação cruzada utilizando as métricas de distância. Observando o quadro, nota-se que a métrica Euclidiana e Mahalanobis obtiveram o mesmo valor de acurácia e desvio, no entanto, a métrica Euclidiana obteve um

menor tempo de processamento. Com isso conclui-se que das métricas comparadas, os modelos que obtiveram melhor desempenho foram as que utilizam a métrica Euclidiana.

Quadro 3 - Resultados do experimento utilizando métricas de similaridade

Métrica de Distância	Acurácia	Desvio Padrão	Tempo de processamento [s]
Euclidiana	0,915	0,151	0,067
Manhattan	0,897	0,164	0,066
Chebyshev	0,709	0,399	0,074
Mahalanobis	0,915	0,151	0,129

Fonte: Autor

5.5 Experimentos utilizando DT

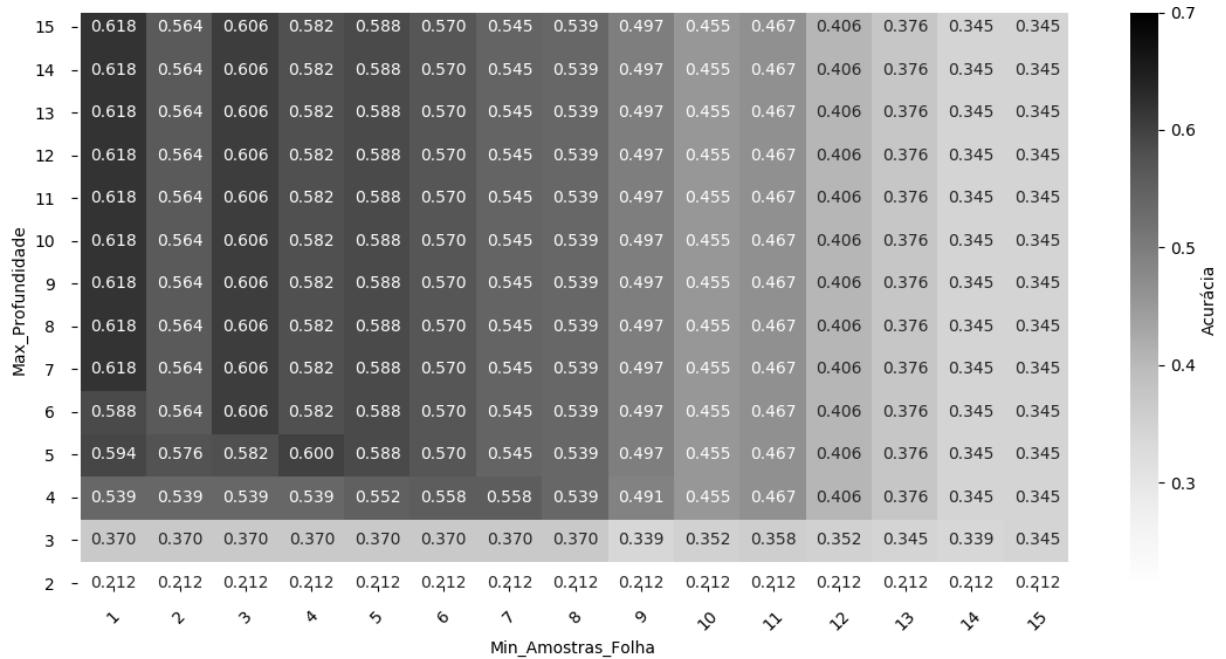
Para a realização dos experimentos de classificação das características obtidas pelo PCA com o método de DT, utilizou-se da classe *DecisionTreeClassifier* contida no pacote *sklearn.tree*, processa o algoritmo CART que constrói árvores binárias. Esta classe possui o parâmetro *criterion* para especificação do critério que mede a qualidade de uma divisão na DT. Além disso, é possível especificar: a profundidade máxima da DT por meio do parâmetro *max_depth*, o número mínimo de amostras necessárias para estar em um nó folha por meio do parâmetro *min_samples_leaf*.

O método de DT foi testado com dois critérios: Entropia e Gini. Para cada critério, testou-se os modelos com variadas combinações dos parâmetros *max_depth* e *min_samples_leaf* da seguinte forma:

- Testaram-se 210 modelos variando o parâmetro *max_depth* em um intervalo de 2 a 15 com passo de 1, e variando o parâmetro *min_samples_leaf* em um intervalo de 1 a 15 com passo de 1.

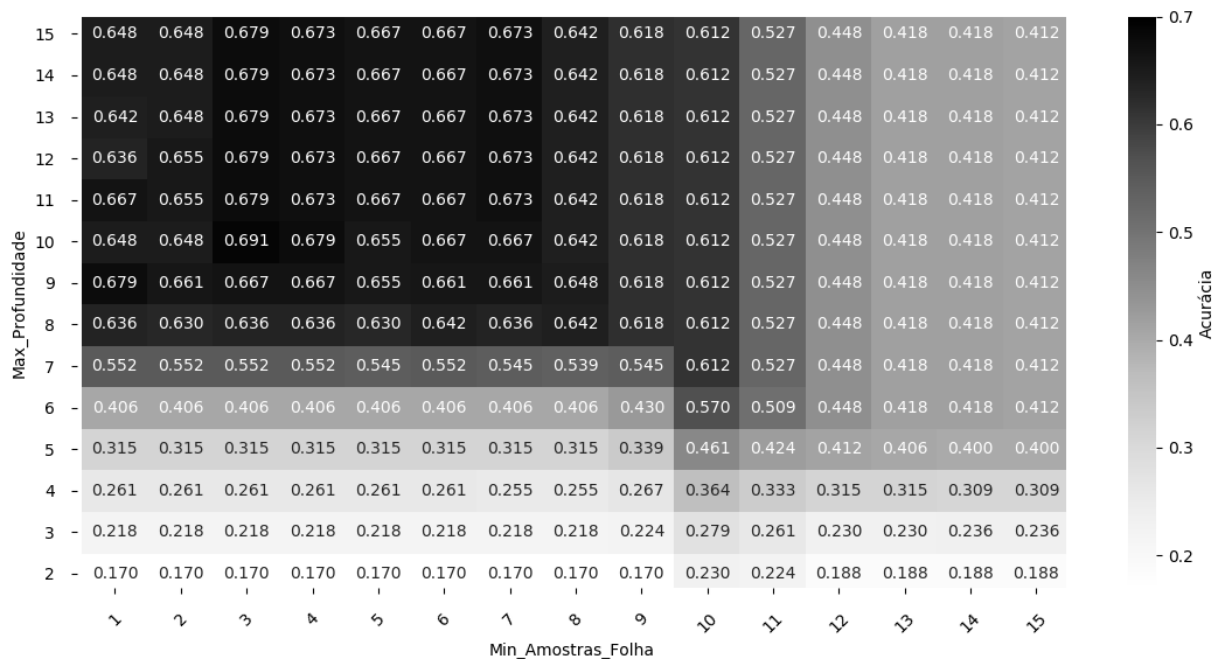
Nas Figura 18 e Figura 19 são mostrados mapas de calor, os quais, apresentam a taxa de acurácia em função da variação dos parâmetros *max_depth* (eixo horizontal) e *min_samples_leaf* (eixo vertical), obtidos em cada teste realizado com os critérios Entropia e Gini respectivamente. Estes mapas de calor são compostos por 210 retângulos que representam cada modelo testado, quanto mais escuro o mesmo, maior a taxa de acurácia do modelo.

Figura 18 - Acurácia dos modelos de DT obtidos pela variação dos parâmetros Min_Amostras_Folha e Max_Profundidade com critério Entropia



Fonte: Autor

Figura 19 - Acurácia dos modelos de DT obtidos pela variação dos parâmetros Min_Amostras_Folha e Max_Profundidade com critério Gini



Fonte: Autor

Analisando esses mapas de calor, observa-se que os modelos treinados com DT não obtiveram bom desempenho quando comparados aos outros classificadores. A principal causa do baixo desempenho seria devido à baixa capacidade de generalização das classes para o

problema de FR. Os melhores modelos são mostrados no Quadro 4. Nesse quadro é mostrado o critério, *max_depth*, *min_samples_leaf*, acurácia média obtida pela validação cruzada, desvio padrão da acurácia e o tempo de processamento. Por meio das informações contidas nesse quadro, é notório que o modelo com critério Gini foi melhor em relação ao de Entropia devido ao maior valor de acurácia e menor tempo de processamento. Foram realizados testes variando a profundidade máxima para valores maiores que 15, porém não apresentou resultados relevantes, assim foi preferível não os apresentar neste trabalho.

Quadro 4 - Melhores modelos obtidos com DT para cada critério utilizado

Critério	Profund. Máx.	Mín. Am. Folha	Acurácia	Desvio padrão	Tempo de processamento [s]
Entropia	10	1	0,618	0,238	0,205
Gini	10	3	0,691	0,237	0,111

Fonte: Autor

No Quadro 5 é mostrado o resultado médio de todos os modelos DT testados com cada critério utilizado, onde pode ser visualizado a acurácia média, desvio padrão da acurácia, tempo de processamento médio e desvio padrão do tempo de processamento. Esse quadro contribui para compreender qual Critério apresentou melhores resultados gerais e menor dependência de ajuste dos parâmetros para obtenção de um modelo com resultados satisfatórios. Observando esse quadro nota-se que os modelos de critérios Gini e Entropia obtiveram valores de acurácia média relativamente próximos, porém o critério Gini demonstrou maior dispersão nos valores de acurácias. O maior diferencial observado é que os modelos treinados com critério Gini apresentaram tempo de processamento aproximadamente 2,5 vezes menor que o de critério Entropia.

Quadro 5 - Resultado médio de todos os modelos DT testados com cada critério utilizado

Critério	Acurácia Média	Desvio da acurácia média	Tempo médio de processamento [s]	Desvio do tempo médio [s]
Entropia	0,468	0,116	0,259	0,110
Gini	0,476	0,166	0,097	0,022

Fonte: Autor

5.6 Experimentos utilizando SVM

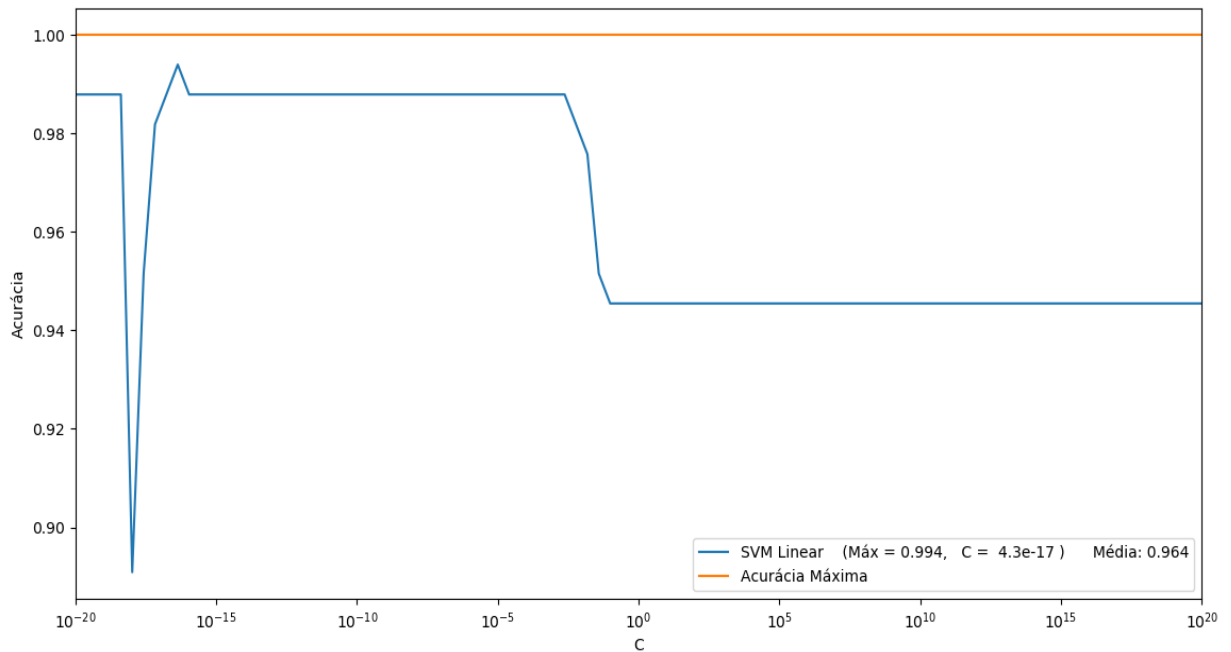
Para a realização dos experimentos de classificação das características obtidas pelo PCA com o método SVM, utilizou-se da classe *SVC* contida no pacote *sklearn.svm*. Esta classe possui um parâmetro para a especificação do *kernel*, além disso é possível especificar os parâmetros que influenciam no cálculo do *kernel* escolhido. A classe *SVC* também oferece suporte a multiclasse empregando a técnica OAO explicada na Seção 2.8.3.

O método SVM foi testado com 4 funções de *kernel*: Linear, RBF, Sigmoidal e Polinomial. Para cada função de *kernel*, testou-se os modelos com variadas combinações dos seus respectivos parâmetros da seguinte forma:

- *Kernel* Linear: Testaram-se 100 modelos variando o parâmetro *C* (Penalidade) em um intervalo com 100 valores igualmente espaçados de 10^{-20} a 10^{20} .
- *Kernel* RBF e Sigmoidal: Testaram-se 100 modelos variando os parâmetros *C* e *Gamma* em um intervalo com 10 valores igualmente espaçados de 10^{-10} a 10^{10} .
- *Kernel* Polinomial: Testaram-se 400 modelos variando os parâmetros *C* e *Gamma* em um intervalo com 10 valores igualmente espaçados de 10^{-10} a 10^{10} e parâmetro *Degree* (grau do polinômio) com valores inteiros em um intervalo de 2 a 5.

Na Figura 20 é mostrado um gráfico dos resultados dos testes realizados com os modelos de SVM de *kernel* Linear. Esse gráfico mostra a taxa de acurácia (eixo vertical) em função da variação do parâmetro *C* (eixo horizontal). Nota-se que os modelos testados no intervalo em que *C* assume valores entre 10^{-17} e 10^{-3} obtiveram as maiores taxas de acurácia. O melhor modelo encontrado apresentou 99,4% de acurácia e foi registrado no Quadro 6, também foi registrado o desvio padrão da acurácia e o tempo de processamento do modelo.

Figura 20 – Taxas de Acurácia dos modelos SVM obtidas pelas variações do parâmetro C do *kernel* Linear



Fonte: Autor

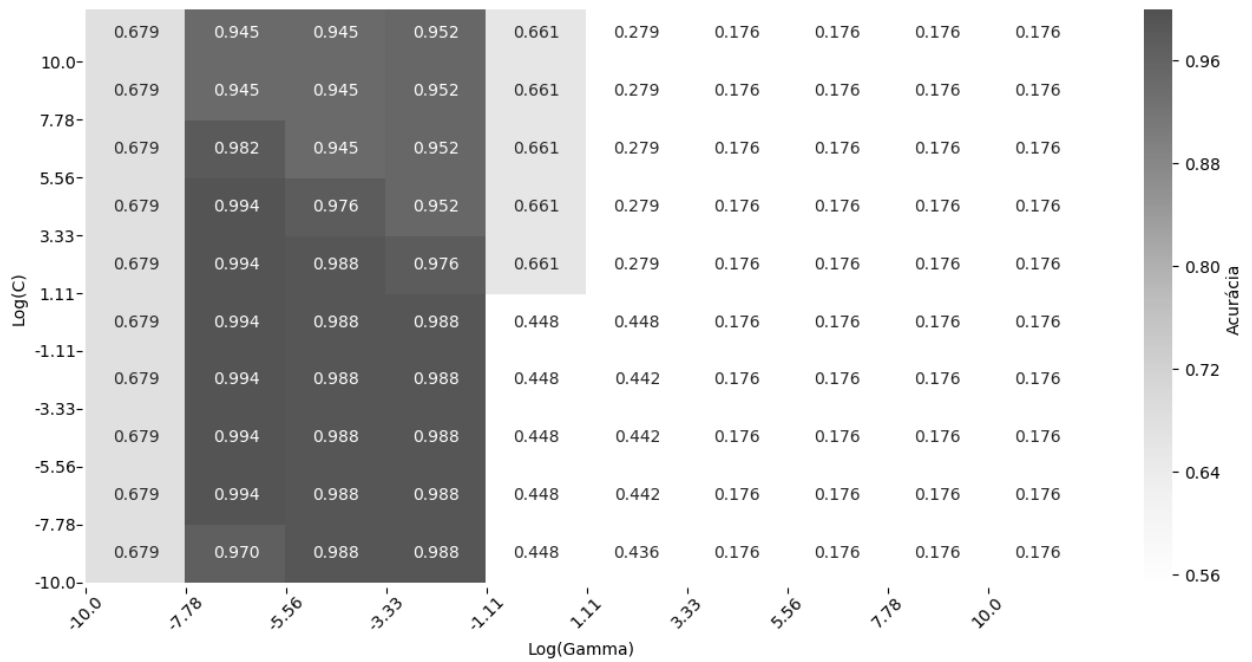
Quadro 6 - Melhor modelo do método SVM utilizando *kernel* Linear

C	Acurácia	Desvio padrão	Tempo de Processamento [s]
$4,33 \times 10^{-17}$	0,994	0,019	0,095

Fonte: Autor

Nas Figura 21 e Figura 22 são mostrados mapas de calor dos resultados dos modelos SVM com *kernel* RBF e Sigmoidal respectivamente. Esses mapas mostram as taxas de acurácia em função da variação dos valores dos parâmetros C (eixo horizontal) e Γ (eixo vertical) em escala logarítmica, obtidos em cada teste realizado. São compostos por 100 retângulos que representam cada modelo testado, quanto mais escuro o retângulo, maior a taxa de acurácia do modelo.

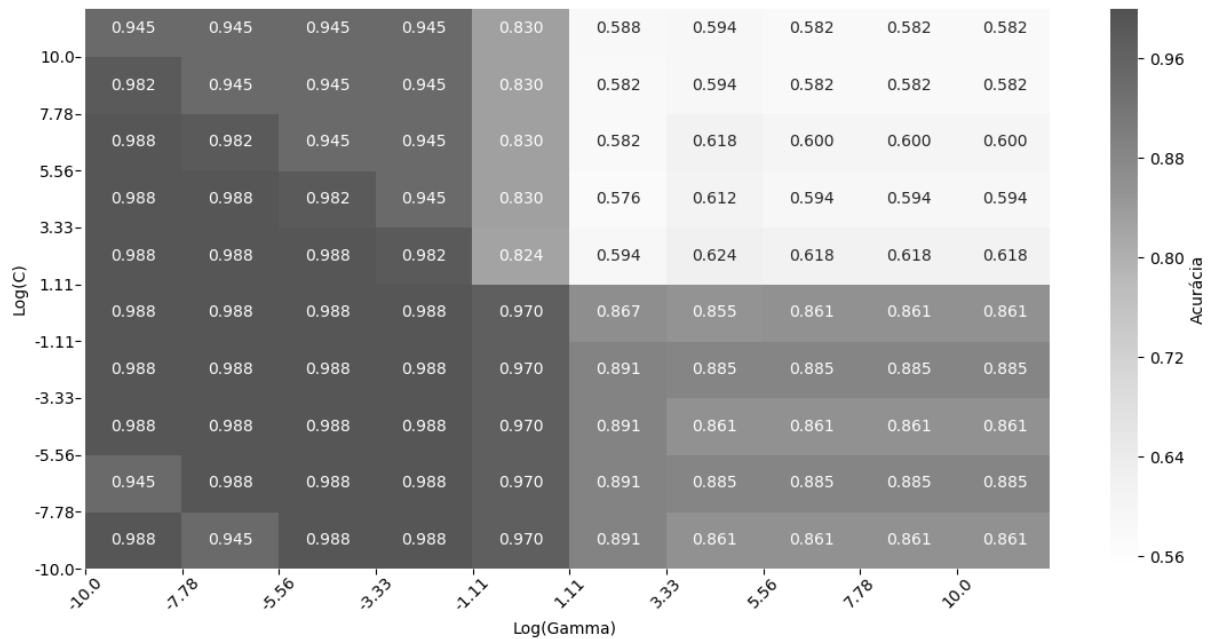
Figura 21 - Acurácia dos modelos SVM com a variação dos parâmetros C e Gamma do *kernel* RBF



Fonte: Autor

Analisando o gráfico do SVM de *kernel* RBF mostrado na Figura 21, observa-se que no intervalo em que *Gamma* assume valores entre $10^{-7.78}$ e $10^{-1.11}$ obteve-se as maiores taxas de acurácia. Fora desse intervalo os modelos testados não apresentaram taxas relevantes. Outra observação é que a variação do parâmetro *C* não provocou variações tão significativas quanto a de *Gamma*. O melhor modelo utilizando *kernel* RBF levando em consideração os critérios citados na Seção 3.4 foi registrado no Quadro 7.

Figura 22 - Acurácia dos modelos SVM com a variação dos parâmetros C e Γ do *kernel* Sigmoial



Fonte: Autor

Quadro 7 - Melhor modelo do método SVM utilizando *kernel* RBF

C	Gamma	Acurácia	Desvio padrão	Tempo de processamento [s]
$1,67 \times 10^{-8}$	$1,67 \times 10^{-8}$	0,994	0,019	0,111

Fonte: Autor

Analisando o gráfico do SVM de *kernel* Sigmoial na Figura 22, observa-se que no intervalo em que Γ assume valores entre 10^{-10} e $10^{1.11}$ e C assume valores entre 10^{-10} e 10^{10} obteve-se as maiores taxas de acurácia, fora desse intervalo os modelos testados não apresentaram taxas relevantes. A variação do parâmetro C apresentou variações tão significativas quanto ao de Γ . O melhor modelo levando em consideração os critérios citados na Seção 5.3 foi registrado no Quadro 8.

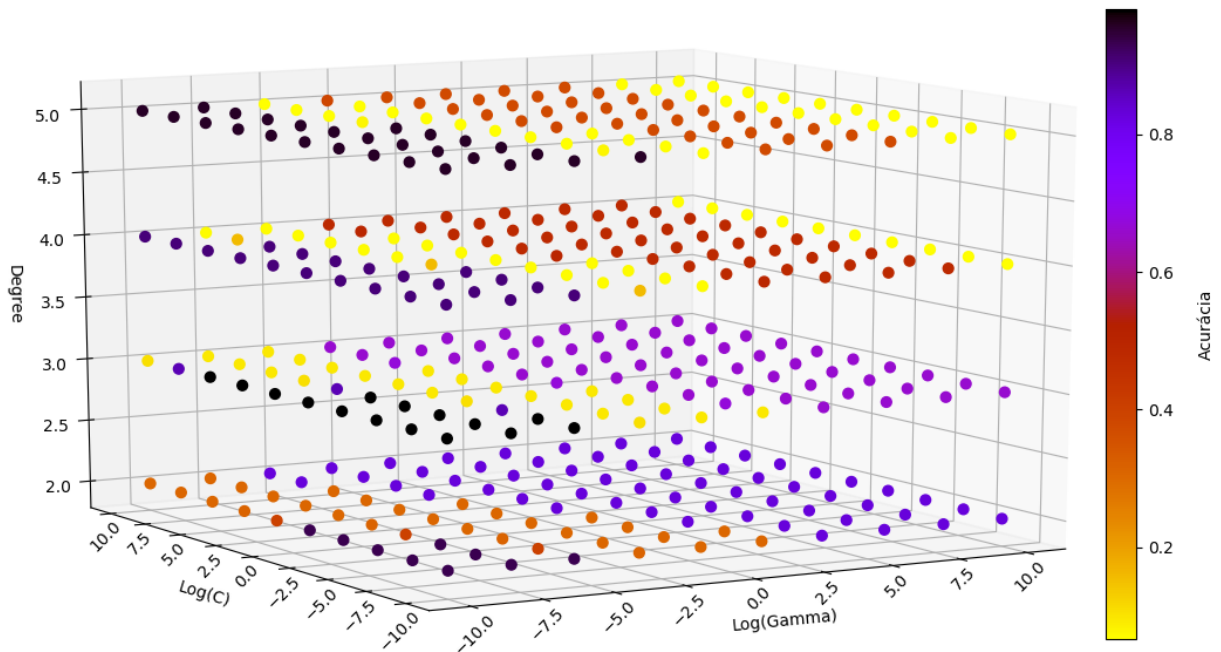
Quadro 8 - Melhor modelo do método SVM utilizando *kernel* Sigmoial

C	Gamma	Acurácia	Desvio padrão	Tempo de processamento [s]
$77,4 \times 10^{-3}$	10^{-10}	0,988	0,038	0,091

Fonte: Autor

Na Figura 23 é mostrado um gráfico em três dimensões, o qual apresenta a acurácia em função da variação dos valores dos parâmetros C , Γ e $Degree$ (grau do polinômio) obtidos em cada teste realizado com *kernel* polinomial. Este gráfico é composto por 400 pontos que representam cada modelo testado. A taxa de acurácia varia com a cor dos pontos, a graduação dos pontos é dada por uma barra de cor, em que, os valores máximos são próximos à cor preta e os valores mínimos próximos à cor amarela. Os eixos horizontais demonstram a variação dos parâmetros C e Γ em escala logarítmica, já o eixo vertical demonstra a variação do parâmetro $Degree$.

Figura 23 - Acurácia dos modelos SVM com a variação dos parâmetros C , Γ e $Degree$ do *kernel* Polinomial



Fonte: Autor

Analisando o gráfico do SVM de *kernel* polinomial na Figura 23 observa-se que nos polinômios de grau 4 e 5 as taxas de acurácias dos modelos testados são distribuídas com valores dispersos. Já nos polinômios de grau 2 e 3 observou-se mais uniformidade na distribuição dos valores das acurácias, comparado aos modelos de grau mais elevado. O melhor modelo levando em consideração os critérios citados na Seção 5.3 foi registrado no Quadro 9.

Quadro 9- Melhor modelo do método SVM utilizando *kernel* Polinomial

C	Gamma	Degree	Acurácia	Desvio padrão	Tempo de processamento [s]
3×10^{-6}	$1,67 \times 10^{-8}$	3	0,982	0,030	0,086

Fonte: Autor

No Quadro 10 é possível visualizar os resultados médios obtidos de todos os modelos testados com cada *kernel*. Os resultados mostrados são a acurácia média, desvio padrão da acurácia média, tempo médio de processamento e o desvio do tempo médio. Esse quadro contribui para compreender qual *kernel* apresentou melhores resultados gerais e menor dependência de ajuste dos parâmetros para obtenção de um modelo com resultados satisfatórios. Observando o mesmo, nota-se que o *kernel* Linear foi o que apresentou maior acurácia média, menor desvio e menor tempo médio de processamento. Já os *kernels* RBF e Polinomial demonstraram valores de acurácia mais dispersos, fazendo com que seja necessário um ajuste fino dos parâmetros do *kernel* para obtenção de resultados mais eficientes.

Quadro 10 – Resultado médio de todos os modelos SVM testados com cada função de *kernel* testado

Kernel	Acurácia Média	Desvio da acurácia média	Tempo médio de processamento [s]	Desvio do tempo médio [s]
Linear	0,964	0,022	0,112	0,017
Sigmoidal	0,847	0,154	0,115	0,039
RBF	0,522	0,343	0,140	0,033
Polinomial	0,535	0,317	0,125	0,063

Fonte: Autor

Comparando os melhores modelos obtidos de cada *kernel* contidos nos Quadros 6, 7, 8 e 9, visualiza-se que todos os *kernels* testados obtiveram modelos que bastante otimizados, porém, os modelos com *kernel* Linear demonstraram no geral maiores taxas de acurácia, e seu modelo otimizado apresentou acurácia de 99,4 % com baixo desvio padrão e menor tempo de processamento. Desta forma escolheu-se esse modelo o mais adequando dentre os *kernels* testados.

5.7 Experimentos utilizando RNA

Este experimento tem como objetivo a avaliação do desempenho das RNAs na tarefa de FR, tendo como entrada os vetores de características extraídos do banco *Yale Face Database* pelo método PCA. São testadas várias redes com arquiteturas distintas para a compreensão de como os resultados são afetados. Cada modelo de RNA testado possui 45 neurônios na camada de entrada e 15 na de saída da rede. Essa quantidade de neurônios na camada de entrada é devido à quantidade de atributos contidos no vetor de características correspondente a cada imagem de face. Já a quantidade de neurônios na camada de saída é equivalente à quantidade de classes contidas no banco de dados faciais.

Em uma RNA, a quantidade de camadas ocultas possui influência na geometria do hiperplano de decisão. Por exemplo, em uma RNA com apenas uma camada oculta (HL - *Hidden Layer*), cria-se um hiperplano. Redes com duas HLs combinam hiperplanos para formar áreas de decisão convexas. Redes com três HLs combinam hiperplanos convexas para formar hiperplanos de decisão convexas com regiões côncavas. Aumentando o número de HLs permite-se um equilíbrio entre generalização e ajuste fino [47]. Neste trabalho, realizou-se os experimentos em RNAs com uma, duas e três HLs, com diferentes funções de ativação (AF – *Activate Function*) e quantidades de neurônios. As AFs testadas foram as seguintes: Identidade, Logística, ReLU e Tangente hiperbólica. Para a especificação dos modelos das RNAs, utilizou-se a classe *MLPClassifier* contida no pacote *sklearn.neural_network* da biblioteca *scikit-learn*. Esta classe permite dimensionar a quantidade de HLs e neurônios artificiais contidos em cada HL por meio do parâmetro *hidden_layer_sizes*. A AF pode ser especificada no parâmetro *activation*. A classe também permite especificar o algoritmo de otimização da RNA por meio do parâmetro *solver* o qual foi definido como “*lbfgs*” em todos os experimentos. O “*lbfgs*” é recomendado pela biblioteca para conjuntos de dados relativamente pequenos, pois pode convergir mais rapidamente e ter um desempenho melhor. Existem outros parâmetros que podem contribuir no resultado dos modelos das RNAs, porém, nos experimentos foram utilizados os valores padrões.

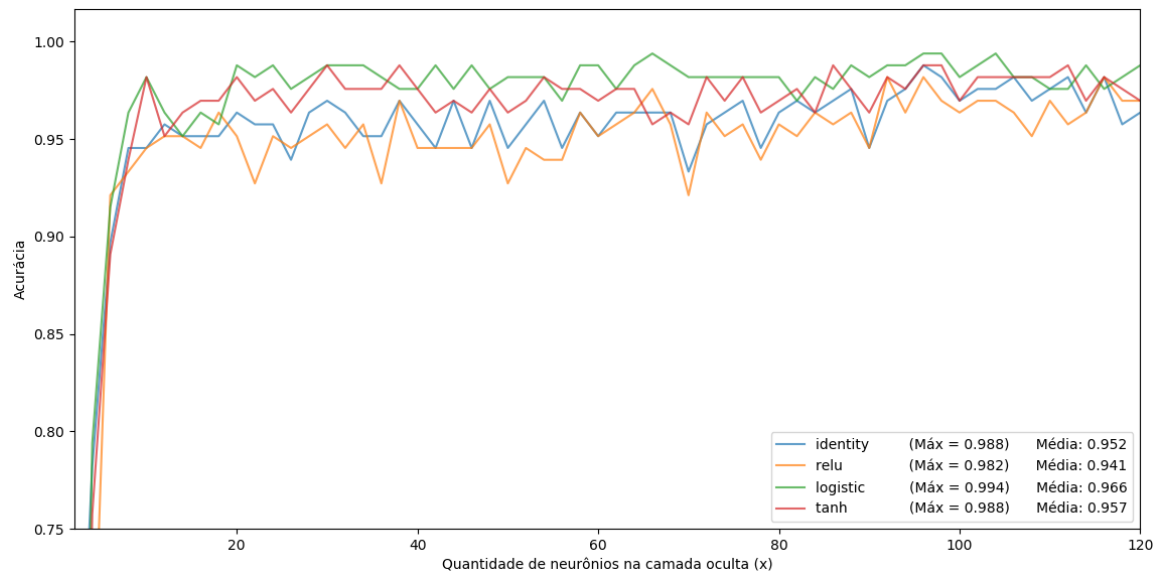
Uma das partes que podem ser mais demoradas na elaboração da arquitetura de um modelo RNA eficiente está em especificar a quantidade de neurônios artificiais de cada HL. Isso devido às diversas possibilidades de combinação da estrutura de neurônios. A complexidade tende a aumentar a medida em que é inserido novas HLs [48]. Quanto maior a quantidade de neurônios na rede, resulta em um período mais longo de treinamento [47]. Assim o ideal é encontrar um modelo que utilize o mínimo de neurônios possível atingindo um alto

poder de generalização. Existem diversas heurísticas para estimar a quantidade desses neurônios, contudo elas não garantem eficácia pois não utilizam conhecimento do domínio dos dados, mesmo assim é uma boa forma de se limitar um certo campo de busca inicial. Algumas dessas heurísticas baseiam-se em usar 75% da quantidade de neurônios da camada de entrada, 50% da quantidade nós de entrada e saída ou usar $2n+1$ neurônios na HL, dado que n é a quantidade de nós na camada de entrada [47]. Sendo assim, baseando-se nessas heurísticas, juntamente com a possibilidade redução do tempo de treinamento de todos modelos, a variação da quantidade de neurônios foi organizada da seguinte maneira: Para modelos com:

- Uma HL: Testaram-se 60 modelos variando a quantidade de neurônios da HL em um intervalo entre 2 e 120 neurônios igualmente espaçados.
- Duas HLs: Testaram-se 64 modelos variando a quantidade de neurônio das HLs em um intervalo entre 20 e 90 neurônios com passo de 10.
- Três HLs: Testaram-se 256 modelos variando a quantidade de neurônios das HLs em um intervalo entre 20 e 90 neurônios com passo de 10.

Na Figura 24 é mostrado um gráfico de taxa de acurácia (eixo vertical) em função da variação da quantidade de neurônios (eixo horizontal) para os modelos de RNAs testadas com apenas uma HL e diferentes AFs. Observando este gráfico, nota-se que a AF logística predominou com maior acurácia na maioria das configurações de HL dos modelos testados, além de que, seu melhor modelo foi obtido com a menor quantidade de neurônios, comparando-se com os melhores modelos das demais AFs. No Quadro 11 são mostrados os melhores modelos obtidos com uma HL para cada AF. Nesse quadro visualiza-se que uma desvantagem da AF Logística foi o tempo de processamento elevado em relação às demais.

Figura 24 - Acurácia de modelos RNA com uma HL em função do número de neurônios artificiais e AF.



Fonte: Autor

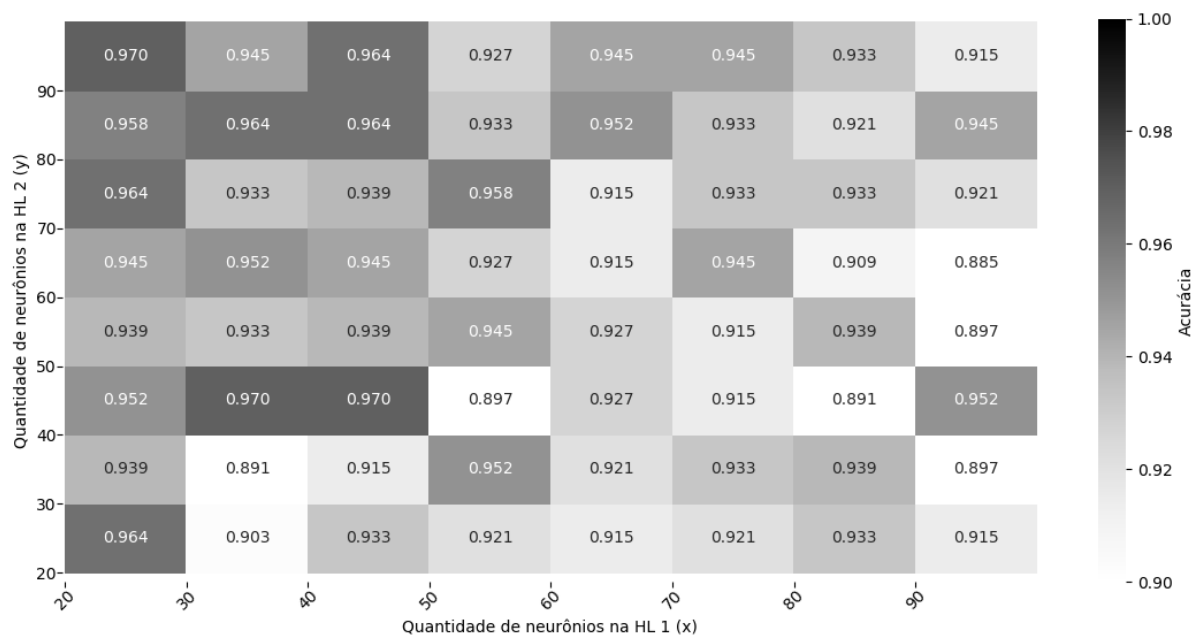
Quadro 11 - Melhores modelos de RNA obtidos com uma HL

AF	HL	Acurácia	Desvio padrão	Tempo de processamento [s]
Identidade	96	0,988	0,026	0,247
Logística	66	0,994	0,019	0,665
ReLU	96	0,982	0,041	0,410
Tanh	112	0,988	0,026	0,259

Fonte: Autor

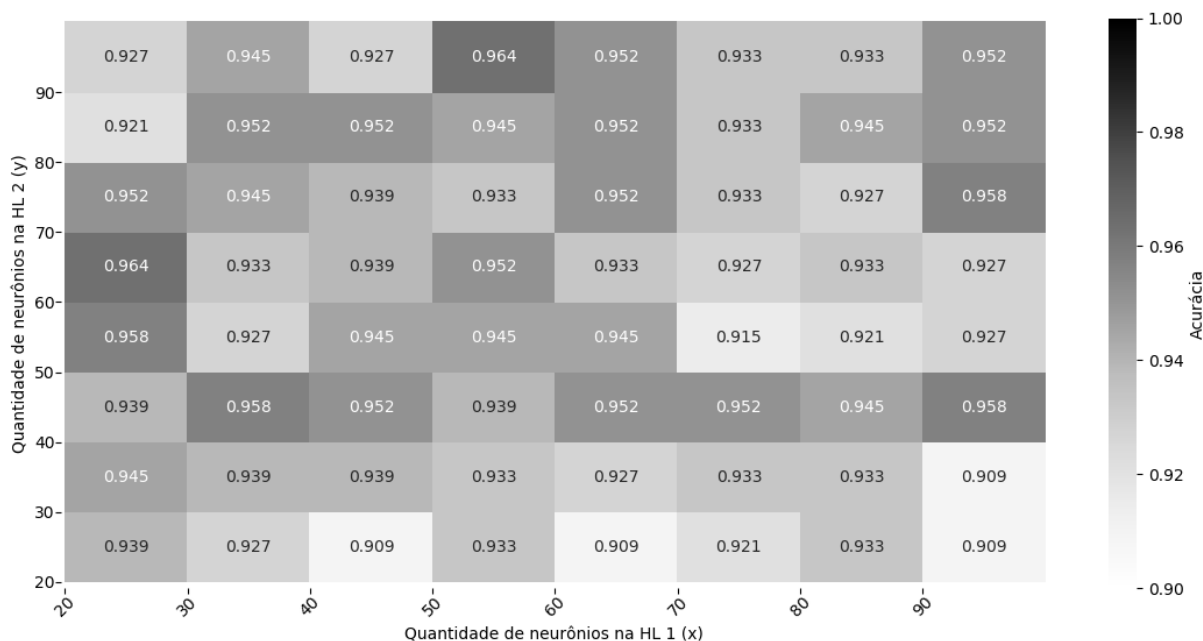
Nas Figuras 25, 26, 27, 28 são mostrados mapas de calor que apresentam a taxa de acurácia em função da variação da quantidade de neurônios artificiais na primeira (eixo horizontal) e segunda (eixo vertical) HL. Cada Figura representa o mesmo teste alterando-se apenas a AF utilizada. Esses mapas de calor são compostos por 64 retângulos que representam cada modelo testado. Quanto mais escuro o mesmo, maior a taxa de acurácia do modelo. Observando essas Figuras, nota-se novamente que a AF Logística obteve maiores taxas de acurácia em relação aos outros modelos com diferentes AFs.

Figura 25 - Acurácia dos modelos de RNAs com duas HLs a partir da variação da quantidade de neurônios artificiais de cada HL utilizando AF ReLU.



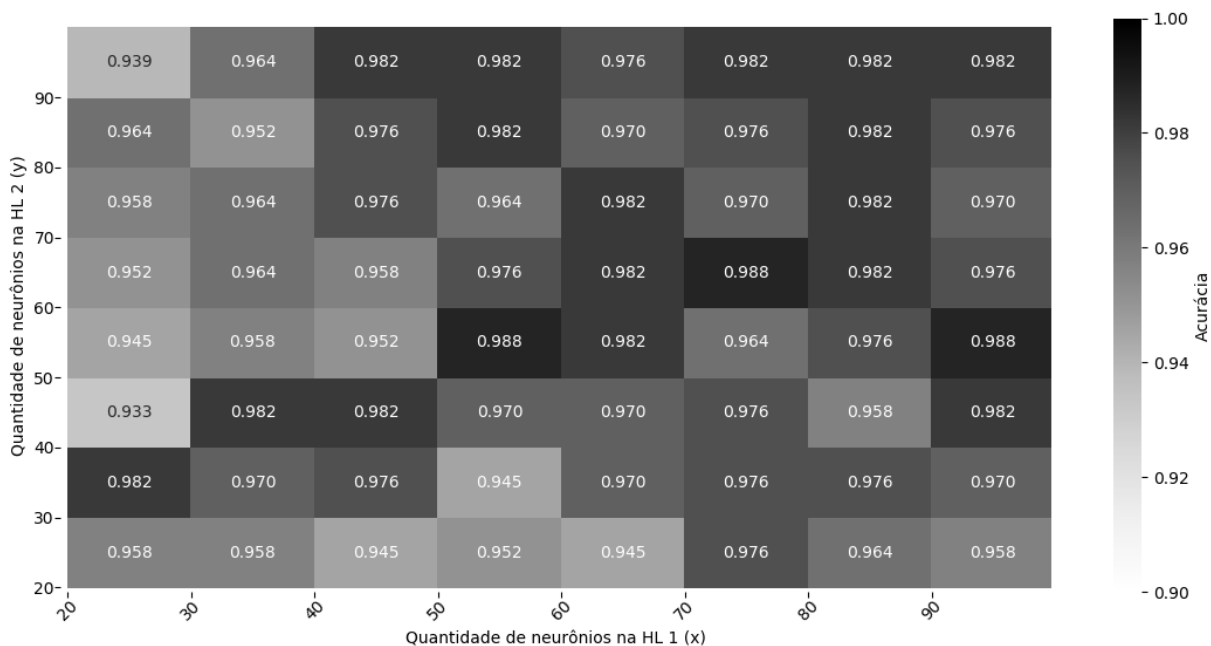
Fonte: Autor

Figura 26- Acurácia dos modelos de RNAs com duas HLs a partir da variação da quantidade de neurônios artificiais de cada HL utilizando AF Identidade



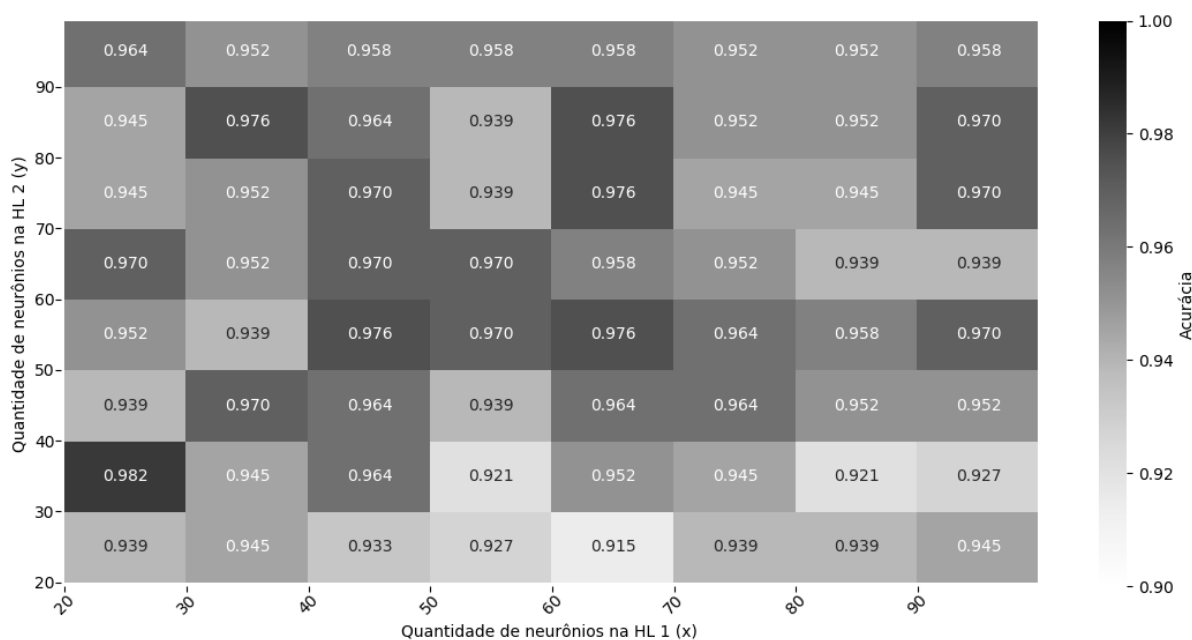
Fonte: Autor

Figura 27- Acurácia dos modelos de RNAs com duas HLs a partir da variação da quantidade de neurônios artificiais de cada HL utilizando AF Logística



Fonte: Autor

Figura 28- Acurácia dos modelos de RNAs com duas HLs a partir da variação da quantidade de neurônios artificiais de cada HL utilizando AF Tanh



Fonte: Autor

No Quadro 12 são mostrados com detalhes os melhores modelos obtidos por cada AF. Nota-se que o melhor modelo foi obtido com AF Logística que apresentou maior taxa de acurácia, menor desvio, porém foi o que gastou mais tempo de processamento. Vale ressaltar que o modelo com AF Tanh também foi bastante eficiente, pois utilizou a menor quantidade de

neurônios nas HLs, também apresentou a segunda maior acurácia e segundo menor tempo de processamento do teste.

Quadro 12 - Melhores modelos de RNA obtidos com duas HLs

AF	HL	Acurácia	Desvio padrão	Tempo de processamento [s]
Identidade	(60, 20)	0,964	0,050	0,281
Logística	(50, 50)	0,988	0,026	1,440
ReLU	(90, 20)	0,970	0,044	1,060
Tanh	(30, 20)	0,982	0,041	0,433

Fonte: Autor

No Quadro 13 são mostrados os melhores modelos obtidos com três HLs para cada AF. Nesse quadro visualiza-se que as AFs Logística, ReLU e Tanh apresentaram mesmo valor de acurácia. A AF ReLU apresentou menor quantidade de neurônios e o segundo maior tempo para o processar os testes. O modelo com AF identidade foi o mais rápido, mas foi o que demonstrou menor valor de acurácia.

Quadro 13 - Melhores modelos de RNA obtidos com três HLs

AF	HL	Acurácia	Desvio padrão	Tempo de processamento [s]
Identidade	(80, 70, 40)	0,964	0,077	0,625
Logística	(70, 70, 80)	0,982	0,030	3,854
ReLU	(40, 40, 20)	0,982	0,030	1,628
Tanh	(80, 80, 20)	0,982	0,058	0,738

Fonte: Autor

No Quadro 14 são mostrados os resultados médios de todos os modelos obtidos para os testes com uma, duas e três HLs com sua respectiva AF. Os resultados mostrados são a acurácia média, desvio padrão da acurácia média, tempo médio de processamento e o desvio do tempo médio. Esse quadro ajuda na compreensão dos ganhos obtidos quando aumentada a quantidade de HLs para o problema em questão. Percebe-se que o aumento da quantidade de HLs provocou redução da acurácia média praticamente em todos os casos, uma exceção no caso da AF logística que também não apresentou um ganho significativo. Isso pode ser explicado pelo fato

de que, quanto mais HLs são adicionadas, o modelo de RNA fica mais sobreajustado aos dados de treinamento. Desta forma, apresentando assim, menor poder de generalização.

Comparando os melhores modelos obtidos de cada configuração de HL e AF observados nos Quadros 1111, 12, e 13, visualiza-se que o melhor modelo escolhido foi o com uma HL e AF Logística. O mesmo apresentou acurácia de 99,4 % com baixo desvio padrão e tempo de processamento mediano.

Quadro 14 - Resultado médio de todos os modelos de RNAs testados com cada combinação de AFs com a quantidade de HLs

Função de Ativação	Quantidade de HL	Acurácia Média	Desvio da acurácia média	Tempo médio de processamento [s]	Desvio do tempo médio [s]
Logística	1	0,966	0,091	0,68	0,27
	2	0,969	0,013	1,32	0,47
	3	0,940	0,020	2,77	0,85
Identidade	1	0,952	0,055	0,24	0,13
	2	0,938	0,014	0,46	0,14
	3	0,917	0,021	0,50	0,13
ReLU	1	0,941	0,076	0,38	0,27
	2	0,933	0,021	0,79	0,18
	3	0,926	0,021	2,49	0,67
Tanh	1	0,957	0,096	0,32	0,24
	2	0,953	0,015	0,44	0,11
	3	0,938	0,022	0,62	0,14

Fonte: Autor

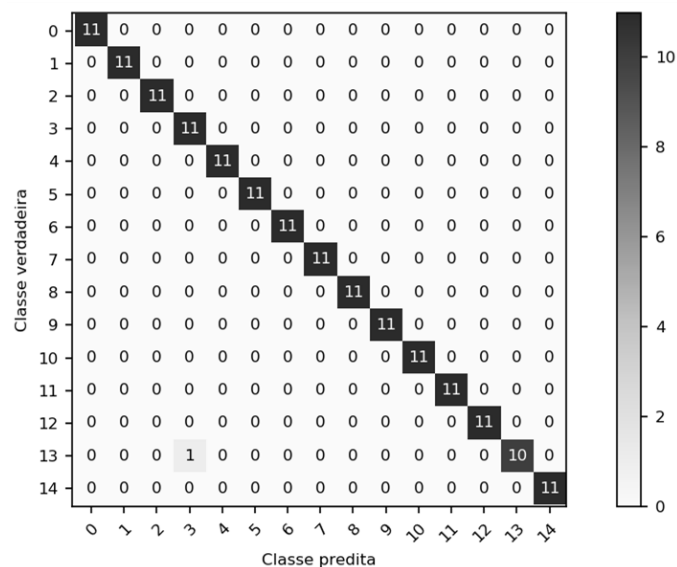
5.8 Comparação dos melhores modelos de cada técnica utilizada

Nessa Seção é abordada uma comparação mais aprofundada com os classificadores que obtiveram melhores resultados nos experimentos realizados nas seções anteriores. Por meio dos experimentos anteriores, foi possível notar que as métricas de similaridade e os classificadores de DT não apresentaram um desempenho tão satisfatório quanto os classificadores de SVM e

RNA. Desta forma as análises e avaliações realizadas nessa Seção se restringem apenas aos melhores classificadores de SVM e RNA escolhidos nas seções anteriores. O objetivo dos experimentos aqui descritos são de julgar qual classificador testado possui um maior poder de generalização.

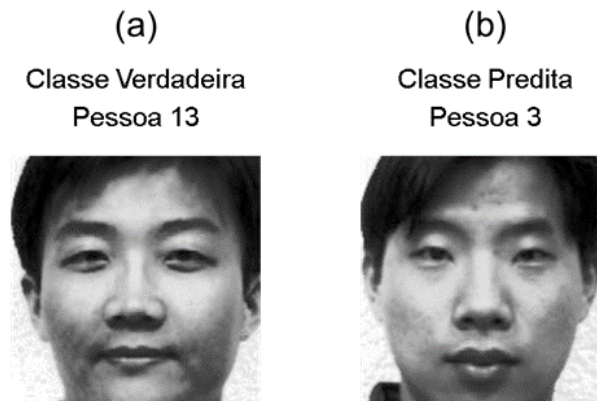
A princípio, calculou-se as matrizes de confusão de ambos os modelos para o entendimento de quais classes foram confundidas nos testes de validação cruzada realizados, desta forma é possível identificar classes que podem ser provedoras de ruídos para um classificador. Para o cálculo da matriz de confusão, utilizou-se do método *confusion_matrix* contido no pacote *sklearn.metrics*. Nesse deve-se informar dois parâmetros: *y_true* e *y_predict* que são as classes verdadeiras e as classes que foram preditas. Esses parâmetros podem ser obtidos pelo cálculo da validação cruzada pelo método *cross_val_predict* contido no pacote *sklearn.model_selection*. A implementação do método *cross_val_predict* é equivalente ao explicado na Seção 5.3. Na Figura 29 é mostrada a matriz de confusão dos melhores modelos de classificador de RNA e SVM (os resultados foram idênticos). No eixo horizontal é mostrado o id da face predita e no eixo vertical o id da face verdadeira. Nota-se que ambos classificadores confundiram a pessoa 3 e 13. Na Figura 30 é mostrada as faces que foram confundidas pelos classificadores. Nessa Figura é possível visualizar uma certa semelhança entre as duas pessoas, devido aos seus traços de descendência asiática, sobrancelhas, nariz e também no formato dos rostos.

Figura 29 - Matriz de confusão do melhor classificador de RNA e SVM, ambas matrizes foram idênticas.



Fonte: Autor

Figura 30 - Comparação das classes (a) verdadeira e (b) predita erroneamente pelos classificadores SVM e RNA



Fonte: Autor

Por fim, realizou-se um último experimento com o objetivo de verificar se os métodos RNA e SVM selecionados, são eficientes na classificação de um conjunto de dados, que contém mais classes e variações de aparência. Para esse experimento, escolheu-se o banco de dados de faces *ORL Database of faces* [49], constituído de 400 imagens faciais de 40 indivíduos distintos sendo que 36 são homens e 4 mulheres. Cada indivíduo possui 10 amostras de imagens faciais com variações de expressões, iluminação, poses e tempo. Todas as imagens da base possuem dimensões padronizadas, cada imagem sendo uma matriz de tamanho 96×112 pixels em escala de cinza de 256 níveis possuindo um total de 10.752 pixels.

Realizou-se a equalização de histograma, segmentação das faces, cálculo do PCA e validação cruzada seguindo os mesmos procedimentos explicados nas Seções 5.1, 5.2 e 5.3. As diferenças foram as seguintes:

- Após a segmentação as imagens foram redimensionadas para o tamanho original de 96×112 pixels.
- O número de componentes necessários correspondentes à 90% da variância do conjunto de dados do banco *ORL Database of faces* foi igual a 98.
- A quantidade k partições (*folds*) no processo de validação cruzada foi definido como igual mesma quantidade de amostras por classe, ou seja, 10.

Nos Quadros 15 e 16 são mostrados os resultados dos testes realizados com os modelos de SVM e RNA. Os resultados mostrados são a acurácia média, desvio padrão da acurácia média e tempo de processamento. Observando esses quadros vemos que ambos os classificadores apresentaram decréscimo nas taxas de acurácia em relação as obtidas com a base de dados *Yale Face Database*. No entanto, o modelo de SVM teve um decaimento

maior que o da RNA. Enquanto o de RNA decaiu apenas $\sim 2\%$, o de SVM decaiu $\sim 17\%$. Através desse experimento não é possível afirmar que todos modelos de SVM são piores, devido a possibilidade do ajuste dos parâmetros ótimos para o banco de faces em questão, mas pode-se afirmar que os modelos de RNA apresentaram maior consistência da acurácia com a mudança do conjunto de dados.

Quadro 15 - Resultados obtidos nos testes com o melhor classificador SVM utilizando o banco de faces *ORL Database of faces*.

Kernel	C	Acurácia	Desvio padrão	Tempo de Processamento [s]
Linear	$4,33 \times 10^{-17}$	0,823	0,076	0,438

Fonte: Autor

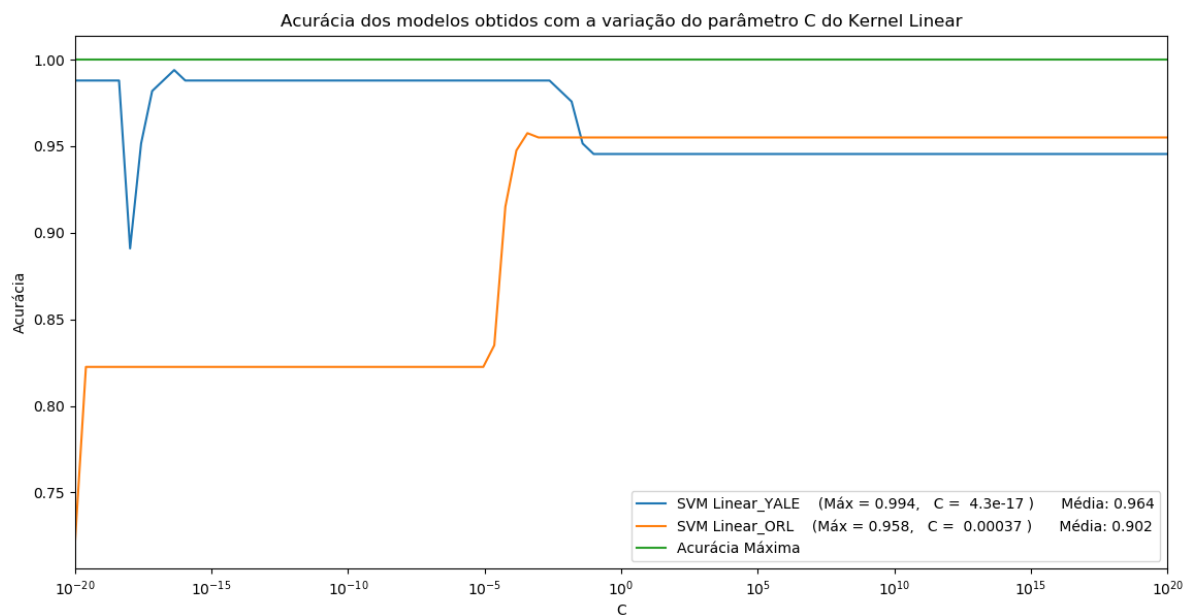
Quadro 16 - Resultados obtidos nos testes com o melhor classificador RNA utilizando o banco de faces *ORL Database of faces*

AF	HL	Acurácia	Desvio padrão	Tempo de processamento [s]
Logística	66	0,968	0,025	1,859

Fonte: Autor

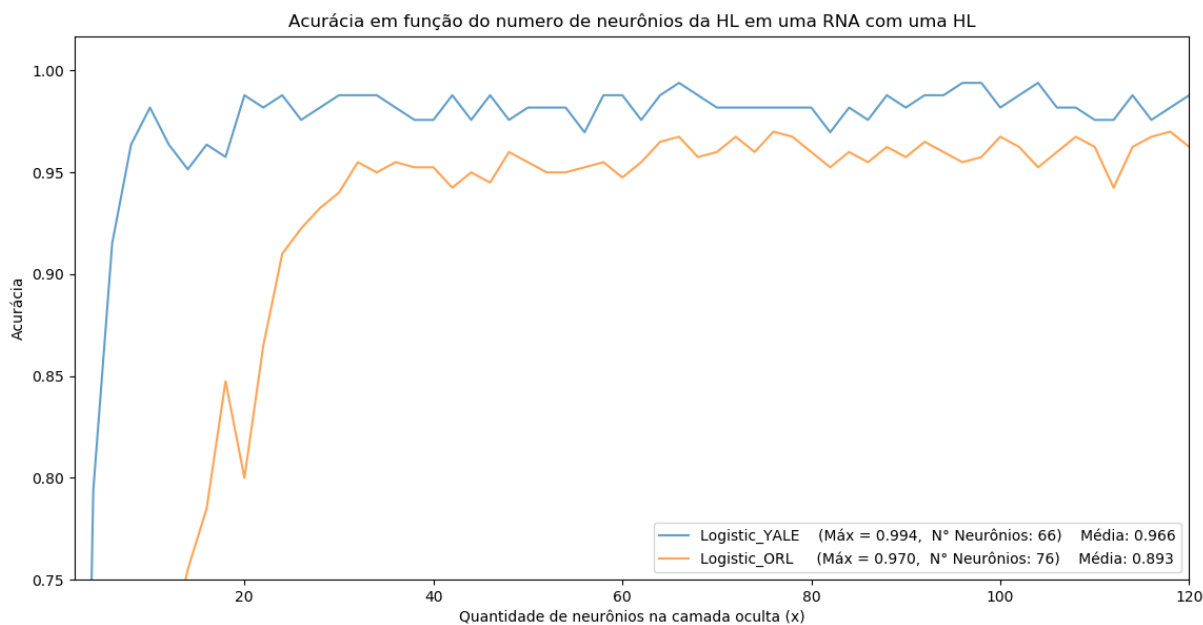
Para confirmar as afirmações anteriores sobre o experimento, realizou-se os mesmos testes apresentados nas Figuras 20 e 24, mas dessa vez utilizando o banco de imagens *ORL Dataset of Faces*. Os resultados são mostrados nas Figuras 31 e 32. Observando a Figura 31 visualiza-se que para valores maiores do parâmetro de penalidade C é possível atingir quase 96% de acurácia. Já na Figura 32 observa-se que tanto para os modelos testados com o banco Yale quanto para o ORL as curvas se demonstraram mais parecidas, diferenciando apenas por um deslocamento lateral e um pouco no nível de amplitude. A acurácia máxima obtida com os modelos de RNA testados com o banco ORL variando a quantidade de neurônios foi de 97%. Com isso percebe-se que mesmo alterando-se a base de dados e realizando um ajuste nos parâmetros dos classificadores SVM e RNA é possível obter modelos com valores de taxas de acurácia bastante significativas.

Figura 31 - Acurácia dos modelos de SVM obtidos com a variação do parâmetro C do *kernel* Linear para os bancos de faces *Yale Face Database* e *ORL Dataset of Faces*



Fonte: Autor

Figura 32 - Acurácia dos modelos de RNAs com uma HL a partir da variação da quantidade de neurônios artificiais utilizando AF Logística com imagens dos bancos de dados *Yale Face Database* e *ORL Dataset of Faces*



Fonte: Autor

Capítulo 6

Considerações Finais

Este trabalho mostra uma metodologia para avaliação e comparação de algoritmos clássicos de classificação aplicados sobre características extraídas de imagens faciais com o objetivo de identificar as pessoas presentes nas imagens. Apresentou-se uma série de técnicas de aprendizado de máquina que foram aplicadas em um contexto de reconhecimento facial. Inicialmente, foi realizada a detecção de faces com os classificadores em cascata. A seguir, efetuou-se a equalização do histograma para a redução da sensibilidade dos resultados a variações de iluminação das imagens. A técnica de componentes principais foi utilizada como extrator de características e por fim a utilização dos classificadores baseados em métricas de similaridade, árvores de decisão, máquina de vetores de suporte e redes neurais artificiais. Diversas combinações de configurações dessas técnicas foram testadas com imagens dos bancos de faces *Yale Face Database*. As configurações com melhores desempenho nesse teste foram aplicadas a imagens do *ORL Database of faces*.

No experimento utilizando as métricas de similaridade como classificador, a métrica Euclidiana obteve 91,5% de acurácia com 1,51 de desvio e menor tempo de processamento dentre todas as métricas avaliadas. Porém o tempo de classificação aumenta com o tamanho do conjunto de treinamento utilizado. A simplicidade de implementação é um fator positivo.

Os classificadores de DT foram os que apresentaram pior desempenho, o melhor modelo de DT obteve 69,1% de acurácia com desvio de 23,7. Esse baixo desempenho pode ser justificado devido ao *overfitting* do conjunto de treinamento, pois a quantidade de amostras (11 por classe) foi insuficiente para uma boa generalização. As DTs com critério Gini demonstraram maiores valores de acurácia em relação aos com Entropia.

Os classificadores SVM foram testados com os *kernels* Linear, Sigmoidal, RBF e Polinomial. Os modelos com *kernel* Linear desempenharam maiores taxas de acurácia e menor tempo de processamento. O melhor modelo de SVM obteve 99,4% de acurácia com desvio de 1,9. Os modelos com os outros *kernels* apresentaram bons resultados, mas com uma necessidade maior no ajuste de parâmetros.

Os classificados de RNA foram testados em configurações com uma, duas e três camadas ocultas variando a quantidade de neurônios artificiais e funções de ativação. Observou-se que os modelos com apenas uma camada oculta foram suficientes para o problema considerado. A melhor RNA obtida utilizou a função Logística e apresentou 99,4% de acurácia

com desvio de 1,9, desempenho equivalente ao do melhor modelo obtido com SVM. Uma desvantagem foi o tempo de treinamento, mas uma vez treinado o modelo, a predição é consideravelmente rápida.

Os modelos ajustados que obtiveram melhores taxas de acurácia foram testados no banco de dados *ORL Database of faces*. Sem efetuar novamente o ajuste dos parâmetros, os dois modelos, um de RNA e um de SVM, foram aplicados às novas imagens. Foram mantidas as mesmas condições presentes nos experimentos anteriores, ou seja, foi efetuada a segmentação, ajuste de histograma, seleção de componentes com 90% de variância e uso da validação cruzada com número de *folds* igual ao número de amostras por classe. Nessas condições, o modelo de RNA obteve melhor resultado, com 96,8 de acurácia com desvio de 2,5.

Observou-se que a utilização da técnica de análise de componentes principais como extrator de características foi muito eficiente. Por meio desta técnica, foi possível representar uma amostra utilizando menos de 1% da quantidade de atributos originais da mesma.

Para trabalhos futuros sugere-se a realização dos experimentos com outros conjuntos de dados que possuam mais variações na aparência tal como o *Labelled Faces in the Wild* citado no Quadro 2. Além disso, sugere-se a utilização de outras técnicas para a classificação das imagens de face, como a Floresta aleatória (*Random Forest*), Redes Neurais Convolucionais e outras técnicas de aprendizado profundo. Também poderia ser verificado a influência do uso de técnicas de normalização da imagem nos resultados. Utilização de outras técnicas para extração de características tal como o Histograma de Gradientes Orientados para fins comparativos à análise de componentes principais.

Referências

- [1] M. C. Monard e J. A. Baranauskas, “Conceitos Sobre Aprendizado de Máquina,” em *Sistemas Inteligentes Fundamentos e Aplicações*, Barueri-SP, Manole Ltda, 2003, pp. 89-114.
- [2] K. Faceli, A. C. Lorena, J. Gama e A. C. P. Leon Ferreira de Carvalho, *Inteligência Artificial : Uma abordagem de aprendizado de máquina*, Rio de Janeiro: Grupo Editora Nacional, 2011.
- [3] M. Wang e W. Deng , “Deep Face Recognition: A Survey,” *arXiv*, pp. 1-26, 2018.
- [4] GaúchaZH, “Ser humano é capaz de lembrar-se de 5 mil rostos,” 10 10 2018. [Online]. Available: <https://gauchazh.clicrbs.com.br/tecnologia/noticia/2018/10/ser-humano-e-capaz-de-lembrar-se-de-5-mil-rostos-cjn3lthlm040e01rxqhn6gb9s.html>. [Acesso em 2019 10 31].
- [5] W. Zhao, R. Chellappa, P. J. Phillips e A. Rosenfeld, “Face Recognition: A Literature Survey,” *ACM Computing Surveys*, vol. 35, nº 4, pp. 399-458, 2003.
- [6] S. Ohlyan, S. Sangwan e T. Ahuja, “A Survey On Various Problems & Challenges In Face Recognition,” *International Journal of Engineering Research & Technology (IJERT)*, vol. 2, nº 6, pp. 2533-2538, 2013.
- [7] M. G. Galterio, S. A. Shavit e T. Hayajneh , “A Review of Facial Biometrics Security for Smart Devices,” *Computers* , vol. 7, nº 37, pp. 1-11, 2018.
- [8] A. K. Jain , P. Flynn e A. A. Ross , *Handbook of Biometrics*, New York, NY : Springer, 2008.
- [9] S. Goel, A. Kaushik e K. Goel, “A Review Paper on Biometrics: Facial Recognition,” *International Journal of Scientific Research Engineering & Technology*, vol. 1, nº 5, pp. 12-17, 2012.
- [10] A. L. Silva, “Redução de características para classificação de imagens de faces.,” MOSSORÓ - RN, 2016.

- [11] S. H., “A comparison based study on biometrics for human recognition,” *International Organization Of Scientific Research Journal of Computer Engineering*, vol. 15, pp. 22-29, 2013.
- [12] S. Z. Li e A. K. Jain, *Handbook of Face Recognition*, Second ed., Series, Ed., Springer, 2004.
- [13] P. Viola e M. Jones, “Rapid Object Detection using a Boosted Cascade of Simple,” em *Computer Vision and Pattern Recognition*, Vancouver, Canada, 2001.
- [14] V. do Amaral, C. Fígaro-Garcia, G. J. F. Gattas e C. E. Thomaz, “Normalização Espacial de Imagens Frontais de Face em Ambientes Controlados e Não-Controlados,” *Periodico Cientifico Eletronico da FATEC Sao Caetano do Sul (FaSCi-Tech)*, 10 2009.
- [15] X. Tan, S. Chen, Z.-H. Zhou e F. Zhang, “Face recognition from as single image per person: A survey,” *Pattern Recognition Society*, 2006.
- [16] I. L. Peixoto Andrezza, “Análise de técnicas de normalização aplicadas ao reconhecimento facial,” João Pessoa, 2015.
- [17] M. Jo, “O Embarcados,” 09 Março 2015. [Online]. Available: <https://www.embarcados.com.br/histogramas-ii/>. [Acesso em 05 Junho 2019].
- [18] S. Theodoridis e K. Koutroubas, *Pattern Recogniton*, Academic Press, 2003.
- [19] M. Turk e A. Pentland , “Eigenfaces for Recognition,” *Journal of Cognitive Neuroscience* , vol. 3, nº 1, pp. 71-86, 1991.
- [20] H. Rady , “Face Recognition using Principle Component Analysis,” *International Journal of Computer Science and Network Security*, vol. 11, nº 10, pp. 134-144, 2011.
- [21] C. S. Fujikawa, “Reconhecimento Facial utilizando Descritores de Textura e Aprendizado Não Supervisionado,” Rio Claro - SP, 2017.
- [22] B. B. Klein, “Machine Learning with Python: Decision Trees in Python,” 2011. [Online]. Available: https://www.python-course.eu/Decision_Trees.php. [Acesso em 2019 10 09].
- [23] M. J. Berry e G. S. Linoff, *Data Mining Techniques*, Indiana : Wiley Publishing, 2004.

- [24] O. M. L. Silva, *Uma Aplicação de Árvores de Decisão, Redes Neurais, KNN para identificação de modelos ARMA Não-Sazonais e Sazonais*, Rio de Janeiro / RJ: PUC-Rio, 2005.
- [25] I. H. Witten e E. Frank, *Data Mining: Pratical Machine Learning Tools and Techiques*, Boston: Morgan Kaufmann Publishers, 2011.
- [26] C. Cortes e V. Vapnik, "Support-Vector Network," *Machine Learning*, vol. 20, nº 3, pp. 273-297, 1995.
- [27] S. Haykin, *Redes neurais princípios e prática*, Porto Alegre: BOOKMAN, 2001.
- [28] T. Hastie, R. Tibshirani e J. Friedman, *The elements of Statistical Learning*, New York: Springer New York Inc, 2001.
- [29] R. G. Negri, *Máquina de vetores de suporte adaptativa ao contexto: formalização e aplicações em sensoriamento remoto*, São José dos Campos, 2013.
- [30] R. Herbrich, *Learning Kernel Classifiers: Theory and Algorithms*, Cambridge, Massachusetts: The MIT Press, 2002.
- [31] C. ., Lorena e C. P. L. F. ., A. de Carvalho, "Uma Introdução às Support Vector Machines," *Revista de Informática Teórica e Aplicada - RITA*, vol. XIV, nº 2, pp. 43-67, 2007.
- [32] A. Nied, *Treinamento de redes neurais artificiais baseado em sistemas de estrutura variável com taxa de aprendizado adaptativa*, Belo Horizonte, 2007.
- [33] L. Fleck, M. H. F. Tavares, E. Eyng, A. C. Helmann e M. A. d. M. Andrade, "Redes Neurais Artificiais: Princípios Básicos," *Revista Eletrônica Científica Inovação e Tecnologia*, vol. 1, nº 13, pp. 47-57, 2016.
- [34] S. M. K. Hasan e T. Chowdhury, *Face Recognition Using Artificial Neural Networks*, 2004.
- [35] Amazon Web Services, "Guia do desenvolvedor: Avaliar a precisão do modelo," Amazon Machine Learning, 02 08 2016. [Online]. Available: https://docs.aws.amazon.com/pt_br/machine-learning/latest/dg/evaluating-model-accuracy.html. [Acesso em 2019 06 06].

- [36] G. C. Cawley e N. L. C. Talbot, “On Over-fitting in Model Selection and Subsequent Selection Bias in,” *Journal of Machine Learning Research*, nº 11, p. 2079–2107, 2010.
- [37] U. C. Vision, “Yale Face Database,” UCSD Computer Vision, 1999. [Online]. Available: <http://vision.ucsd.edu/content/yale-face-database>. [Acesso em 27 07 2019].
- [38] “Python,” Python Software Foundation, [Online]. Available: <https://www.python.org/>. [Acesso em 29 04 2019].
- [39] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot e E. Duchesnay, “Scikit-learn: Machine Learning in Python,” *Journal of Machine Learning Research*, vol. 12, pp. 2825--2830, 2011.
- [40] “OpenCV,” OpenCV team, [Online]. Available: <https://opencv.org/>. [Acesso em 29 04 2019].
- [41] M. Sharif, F. Naz, M. Yasmin, M. A. Shahid e A. Rehman , “Face Recognition: A Survey,” *Journal of Engineering Science and Technology Review* , pp. 1-12, 2017.
- [42] V. Pisarevsky, “Github,” OpenCV, 19 12 2013. [Online]. Available: https://github.com/opencv/opencv/blob/master/data/haarcascades/haarcascade_frontalface_alt.xml. [Acesso em 28 10 2019].
- [43] B. Oliveira, “O que é Análise de Componentes Principais? - Oper,” 11 10 2019. [Online]. Available: <https://operdata.com.br/blog/analise-de-componentes-principais/>. [Acesso em 29 10 2019].
- [44] scikit-learn developers (BSD License), “Support Vector Machines,” ScikitLearn, 2019. [Online]. Available: <https://scikit-learn.org/stable/modules/svm.html>. [Acesso em 1 11 2019].
- [45] scikit-learn developers (BSD license), “Modelos de redes neurais,” ScikitLearn, 2019. [Online]. Available: https://scikit-learn.org/stable/modules/neural_networks_supervised.html#multi-layer-perceptron. [Acesso em 1 11 2019].

- [46] F. C. Santos, “Variações do Método kNN e suas Aplicações na Classificação Automática de Textos,” Instituto de Informática, Universidade Federal de Goiás, Goiânia, 2009.
- [47] S. Walczak e N. Cerpa, “Heuristic Principles For The Design of Artificial Neural Network,” *Information and Software Technology*, vol. 41, nº 2, pp. 109-119, 1999.
- [48] E. Mendes e C. A., “Target recognition using evolutionary neural networks,” em *Proceeding of V Brazilian Symposium on neural networks*, São Carlos - SP, 1998.
- [49] K. Damkhang, “AT&T Database of Faces | Kaggle,” Kaggle, 2019. [Online]. Available: <https://www.kaggle.com/kasikrit/att-database-of-faces>. [Acesso em 17 11 2019].
- [50] C. J. Burgues, “A tutorial on support vector machines for pattern recognition,” *Data Mining and Knowledge Discover*, vol. 2, pp. 121-167, 1998.