

Pedro Biagi Dias

**Detecção de Intrusão em Redes de
Computadores com Aprendizado
Semi-Supervisionado**

Uberlândia – MG

2026

Pedro Biagi Dias

Detecção de Intrusão em Redes de Computadores com Aprendizado Semi-Supervisionado

Universidade Federal de Uberlândia – UFU
Faculdade de Computação – FACOM
Graduação em Sistemas de Informação

Orientador: Prof.^a Elaine Ribeiro de Faria Paiva

Uberlândia – MG

2026

SUMÁRIO

1	INTRODUÇÃO	5
1.1	Objetivo	7
1.1.1	Objetivos Específicos	7
1.2	Organização do Trabalho	7
2	FUNDAMENTAÇÃO TEÓRICA	8
2.1	Sistemas de Detecção de Intrusão	8
2.2	Aprendizado de Máquina	9
2.2.1	Pré-processamento	10
2.2.2	Treinamento	11
2.2.3	Avaliação	13
2.3	Algoritmos Semi-supervisionados Clássicos	14
2.4	Trabalhos Relacionados	15
2.4.1	Tabela Comparativa dos Trabalhos Relacionados	17
2.5	Considerações Finais do Capítulo	18
3	COMPARAÇÃO DE ALGORITMOS SEMI-SUPERVISIONADOS	19
3.1	Visão Geral do Método	19
3.2	Aquisição e Organização dos Dados	20
3.3	Pré-processamento dos Dados	21
3.3.1	Seleção de Atributos	22
3.4	Algoritmos Semi-Supervisionados	23
3.5	Treinamento dos Modelos Semi-Supervisionados	25
3.5.1	Tratamento do Desbalanceamento com <i>SMOTE</i>	27
3.6	Avaliação e Validação dos Resultados	27
4	RESULTADOS DOS EXPERIMENTOS	29
4.1	Base de Dados	29
4.2	Resultado do Pré-Processamento	30
4.3	Resultados do <i>Label Propagation</i> (LPA)	31
4.3.1	Resultados da base de terça-feira	31
4.3.2	Resultados da base de sexta-feira	32
4.4	Resultados do <i>Random Forest</i> com <i>Co-Training</i>	32
4.4.1	Resultados da base de terça-feira	32
4.4.2	Resultados da base de sexta-feira	33
4.5	Análise Comparativa entre os Algoritmos	33

4.6	Supervisionado: Random Forest com 1% Rotulado	34
4.7	Considerações Finais do Capítulo	34
5	CONCLUSÃO	36
5.1	Trabalhos Futuros	37
	REFERÊNCIAS	38

RESUMO

A crescente conectividade das redes de computadores ampliou a superfície de ataque e a frequência de ameaças cibernéticas, tornando os Sistemas de Detecção de Intrusão (IDS) essenciais para a segurança da informação. Um dos maiores desafios da área é a escassez de dados rotulados, já que ataques reais são raros e variáveis, o que limita os modelos supervisionados. Nesse cenário, o aprendizado semi-supervisionado se destaca por aproveitar grandes volumes de dados não rotulados. Este trabalho compara dois algoritmos semi-supervisionados aplicados à detecção de intrusão: o *Label Propagation* (LPA) e o *Random Forest* com *Co-Training*. Os experimentos foram conduzidos sobre a base CIC-IDS2017, realizando ocultações de amostras a fim de simular a escassez de rótulos, por meio de diferentes frações de dados rotulados, com particionamento estratificado. O desempenho foi avaliado por acurácia, precisão, revocação e F1-Score. Os resultados mostraram a superioridade do Co-Training, que manteve altos valores de F1-Score e revocação com apenas 1% de dados rotulados, aliado a baixo custo computacional. Já o LPA mostrou-se sensível ao desbalanceamento, dependente de maior supervisão e com elevado custo computacional. Conclui-se que, para a detecção de intrusão com poucos rótulos, o *Random Forest* com *Co-Training* oferece a melhor relação entre desempenho, robustez e eficiência.

Palavras-chave: Sistemas de Detecção de Intrusão. Aprendizado de Máquina. Aprendizado Semi-Supervisionado. *Label Propagation*. *Co-Training*.

1 INTRODUÇÃO

A globalização e a digitalização acelerada ampliaram de maneira expressiva o número de dispositivos conectados e o volume de dados trafegados em redes de computadores. Segundo o Cisco Annual Internet Report [Systems](#) (2023), o tráfego global ultrapassa diversos zettabytes anuais, impulsionado por serviços em nuvem, IoT e aplicações distribuídas. Esse crescimento acompanha a expansão da Internet das Coisas, destacada no Ericsson Mobility Report [Ericsson](#) (2023), que aponta bilhões de dispositivos conectados em operação em ambientes domésticos, corporativos e industriais. Embora essa conectividade massiva favoreça avanços tecnológicos, ela também aumenta significativamente a superfície de ataque e a probabilidade de eventos maliciosos em escala global. Relatórios internacionais, como o IBM Cost of a Data Breach [Security](#) (2023), mostram que incidentes de segurança resultam em perdas financeiras expressivas, interrupção de serviços e comprometimento de dados sensíveis. Esse cenário hiperconectado e vulnerável evidencia a necessidade de mecanismos avançados de monitoramento e detecção de comportamentos suspeitos, capazes de proteger redes cada vez mais complexas e sujeitas a ameaças constantes. Diante do crescimento contínuo das ameaças cibernéticas, a detecção de intrusões tornou-se um componente essencial para a proteção de redes modernas, tanto no setor público quanto no privado([DIANA; DINI; PAOLINI, 2025](#)).

Nesse contexto, os modelos de Aprendizado de Máquina (*Machine Learning*), subárea da inteligência artificial que capacita os sistemas computacionais a aprenderem e a melhorarem seu desempenho na execução de tarefas específicas a partir da análise de dados, são constantemente usadas como uma das principais técnicas para mitigar esses ataques. Para a detecção de intrusões, esses modelos podem ser divididos, de maneira geral, em aprendizado supervisionado e não supervisionado. Os modelos supervisionados são treinados a partir de um histórico de dados no qual cada exemplo possui um rótulo, já os não supervisionados operam sem a necessidade desses rótulos prévios, buscando aprender com características semelhantes.

No entanto, um dos maiores desafios enfrentados pela área é a dificuldade de rotular grandes volumes de tráfego, já que ataques reais são raros, variáveis e frequentemente desconhecidos ([ZHENG; YANG; WANG, 2023a](#)). Isso compromete diretamente o desempenho de modelos supervisionados, que dependem de conjuntos de dados amplos e atualizados para identificar padrões de ataque. Por outro lado, métodos não supervisionados, embora úteis na detecção de anomalias, estão sujeitos a elevadas taxas de falsos positivos e podem confundir comportamentos incomuns, porém legítimos, com ameaças ([MEIDAN et al., 2023](#)). Nesse contexto, o aprendizado semi-supervisionado surge como uma alternativa

altamente relevante, pois combina a precisão dos modelos supervisionados com a flexibilidade dos não supervisionados, permitindo aproveitar grandes quantidades de dados não rotulados, realidade comum em redes corporativas e ambientes distribuídos (MEIDAN et al., 2023). Estudos anteriores, reforçam que técnicas semi-supervisionadas apresentam melhor capacidade de generalização e maior eficácia na identificação de ataques novos ou pouco representados em bases tradicionais Mvula et al. (2024), o que evidencia sua importância crescente tanto para a sociedade, ao fortalecer mecanismos de defesa cibernética, quanto para a academia, ao abrir novas possibilidades de pesquisa em segurança da informação e aprendizado de máquina.

Os trabalhos relacionados ao tema de detecção de intrusão evidenciam a importância de compreender como diferentes abordagens lidam com ataques cada vez mais sofisticados. Enquanto estudos apontam que modelos supervisionados falham ao generalizar novos ataques ALCÂNTARA (2024) e métodos não supervisionados são suscetíveis a altas taxas de falsos positivos (VIEIRA, 2024), a literatura recente tem focado no aprendizado semi-supervisionado para mitigar a escassez de dados rotulados. Aprofundando essa abordagem, modelos semi-supervisionados combinam características dos paradigmas supervisionado e não supervisionado, permitindo um melhor equilíbrio entre capacidade de generalização e redução de erros. Nesse sentido, propostas como a de Švihrová e Lettner (2020) utilizam autoencoders no dataset NSL-KDD, baseando-se no erro de reconstrução de amostras para identificar anomalias e classificar o tráfego de rede. Avançando na expansão de bases restritas, Sarantos, Violos e Leivadreas (2025) aplica o algoritmo *Label Propagation* para enriquecer o conjunto de treinamento de modelos supervisionados, destacando os desafios da generalização em cenários de avaliação cruzada. Culminando nessa evolução em direção a redes de grande escala, Williams e Qian (2025) investigou estratégias iterativas no dataset CIC-DDoS2019, avaliando especificamente o uso de *self-training* com Regressão Logística e *co-training* com *Random Forest*. Ao alcançar alta acurácia com apenas uma fração dos dados rotulados, esse estudo reforça o potencial dessas técnicas e consolida a base teórica para a investigação comparativa proposta neste trabalho.

Diante desse cenário, torna-se evidente a necessidade de investigar abordagens que superem as limitações dos métodos tradicionais de detecção de intrusão. Nesse sentido, o aprendizado semi-supervisionado destaca-se não apenas como uma alternativa teórica, mas como uma solução prática frente à escassez de dados rotulados e à crescente complexidade das ameaças cibernéticas. Assim, a análise e comparação dessas técnicas mostram-se fundamentais para compreender seu potencial e suas limitações, contribuindo para o desenvolvimento de modelos mais eficazes, capazes de operar em ambientes reais, dinâmicos e com grande volume de dados.

1.1 Objetivo

O objetivo geral deste trabalho é investigar o desempenho de técnicas de aprendizado semi-supervisionado para detecção de intrusões em redes de computadores, por meio da comparação de algoritmos já utilizados na literatura, usando um conjunto de dados representativo e medidas de avaliação apropriadas.

1.1.1 OBJETIVOS ESPECÍFICOS

Para alcançar o objetivo geral, definem-se os seguintes objetivos específicos:

- Investigar os principais algoritmos de aprendizado semi-supervisionado aplicados a sistemas de detecção de intrusão
- Identificar, selecionar e preparar um conjunto de dados relevante para avaliação de técnicas de detecção de intrusão, considerando suas características e limitações
- Definir e aplicar um protocolo experimental adequado para cenários com diferentes níveis de dados rotulados e não rotulados
- Analisar as vantagens, limitações e aplicabilidade das técnicas semi-supervisionadas na detecção de tráfego malicioso

1.2 Organização do Trabalho

Esta monografia está estruturada em outros quatro capítulos além deste de introdução. O Capítulo 2 apresenta a fundamentação teórica necessária para o embasamento dos métodos investigados, englobando também a revisão dos principais trabalhos correlatos que norteiam o estado da arte na área. No Capítulo 3, detalha-se a metodologia adotada, descrevendo as etapas de desenvolvimento, as bases de dados e a configuração dos algoritmos utilizados para atingir os objetivos propostos. O Capítulo 4 é dedicado à exposição dos experimentos realizados e à discussão crítica dos resultados obtidos diante dos cenários avaliados. Por fim, o Capítulo 5 traz as conclusões finais, ressaltando as contribuições alcançadas por esta monografia e sugerindo direcionamentos para trabalhos futuros.

2 FUNDAMENTAÇÃO TEÓRICA

Nesta seção serão apresentados os principais pilares que fundamentam o estudo em questão e como eles sustentam o desenvolvimento do trabalho. Inicialmente, será discutida a importância do tema, bem como os principais desafios enfrentados na área, que motivam a realização deste estudo e reforçam a necessidade de pesquisas no campo. Em seguida, serão abordadas as etapas teóricas necessárias para tratar o problema proposto, contemplando as fases de pré-processamento dos dados, treinamento e avaliação do modelo. Por fim, serão apresentados exemplos de modelos semi-supervisionados frequentemente utilizados na literatura, além da discussão de trabalhos relacionados que contribuem para uma melhor compreensão do tema.

2.1 Sistemas de Detecção de Intrusão

A segurança da informação é uma área crítica na computação dedicada a proteger sistemas e informações contra acessos não autorizados, danos ou alterações não permitidas. De uma forma geral, a segurança se baseia em pilares para mecanismos de segurança, sendo chamados de CID: Confidencialidade (garantir que a informação seja acessível apenas a entidades autorizadas), Integridade (proteger a exatidão e a completude dos dados contra modificações indevidas) e Disponibilidade (assegurar que os serviços e dados estejam acessíveis quando requisitados) ([STALLINGS, 2017](#)).

Contudo, a manutenção dessa tríade é constantemente desafiada por ataques cibernéticos. Uma intrusão, ou ataque, pode ser definida como qualquer tentativa deliberada de comprometer a confidencialidade, a integridade ou a disponibilidade de um sistema. Considerando a realidade de um mundo hiperconectado, a frequência e a gravidade dessas ameaças atingiram patamares críticos. Segundo o relatório *Cost of a Data Breach 2024* da IBM, o custo médio global de uma violação de dados saltou para 4,88 milhões de dólares, o maior aumento já registrado desde o período da pandemia ([IBM Security, 2024](#)).

Ressalta-se, ainda, que essa métrica reflete apenas os incidentes oficialmente notificados. Em muitos casos, ataques bem-sucedidos permanecem não detectados, o que faz com que o impacto real a longo prazo seja ainda mais significativo. Como muitas dessas ameaças modernas operam de forma silenciosa, camuflando-se em meio ao tráfego legítimo, a janela de exposição das organizações aumenta consideravelmente, evidenciando a necessidade do apoio de Sistemas de Detecção de Intrusão (IDS).

Os Sistemas de Detecção de Intrusão (IDS) são responsáveis por monitorar atividades em redes ou sistemas computacionais a fim de identificar comportamentos po-

tencialmente maliciosos, atuando de forma complementar a mecanismos preventivos de segurança (LIPPMANN et al., 2000). De acordo com Scarfone, os Sistemas de Detecção de Intrusão podem ser classificados em *Network-based IDS* (NIDS) e *Host-based IDS* (HIDS), sendo os NIDS amplamente empregados em cenários de análise de tráfego de rede, enquanto os HIDS monitoram e analisam os pacotes que trafegam pela rede em busca de atividades suspeitas, os HIDS operam diretamente nos dispositivos, analisando eventos internos do sistema, como arquivos, processos e registros de log, a fim de identificar possíveis intrusões (Scarfone e Mell (2007)).

Quanto à estratégia de detecção, os IDS podem operar por meio de assinaturas, voltadas à identificação de ataques conhecidos, ou por anomalia, que busca detectar desvios em relação ao comportamento normal do sistema (SOMMER; PAXSON, 2010). Embora essas abordagens sejam eficazes em diferentes contextos, ambas enfrentam limitações diante da evolução constante dos ataques, do desbalanceamento entre classes e da escassez de dados rotulados (ZHENG; YANG; WANG, 2023b).

Diante desses desafios, técnicas de aprendizado de máquina tornaram-se centrais no desenvolvimento de IDS modernos, exigindo processos adequados de extração de conhecimento a partir dos dados analisados. Nesse contexto, etapas como preparação e representação dos dados, extração de padrões com dados parcialmente rotulados e uso do conhecimento extraído para apoiar a detecção de intrusão são fundamentais, especialmente quando se adotam abordagens de aprendizado semi-supervisionado (Chapelle, Schölkopf e Zien (2006)).

2.2 Aprendizado de Máquina

O aprendizado de máquina é uma área da inteligência artificial que estuda métodos computacionais capazes de aprender padrões e realizar previsões a partir de dados, sem que sejam explicitamente programados para cada situação específica (MITCHELL, 1997). Esses métodos têm sido amplamente utilizados em diferentes domínios, especialmente em problemas que envolvem grandes volumes de dados e padrões complexos, como é o caso da detecção de intrusão em redes de computadores.

De modo geral, o aprendizado de máquina engloba diferentes tarefas fundamentais, entre as quais destacam-se a classificação, a regressão e o agrupamento (ALPAYDIN, 2020). Neste trabalho, o foco recai sobre a classificação, cujo objetivo principal é atribuir rótulos categóricos a novas instâncias de dados com base em padrões extraídos de características previamente observadas. No contexto de sistemas de detecção de intrusão, essa abordagem preditiva é essencial para distinguir, com o maior grau de precisão possível, o tráfego de rede legítimo de atividades potencialmente maliciosas.

A Figura 1 mostra o processo de classificação em aprendizado de máquina, que

segue de forma geral, uma sequência de etapas bem definidas. Inicialmente, os dados brutos passam por um processo de pré-processamento, no qual são realizadas operações como limpeza, normalização e seleção de atributos. Em seguida, ocorre a etapa de treinamento, na qual os modelos aprendem padrões a partir dos dados disponíveis. Por fim, os modelos são avaliados por meio de métricas adequadas, permitindo analisar seu desempenho e capacidade de generalização.

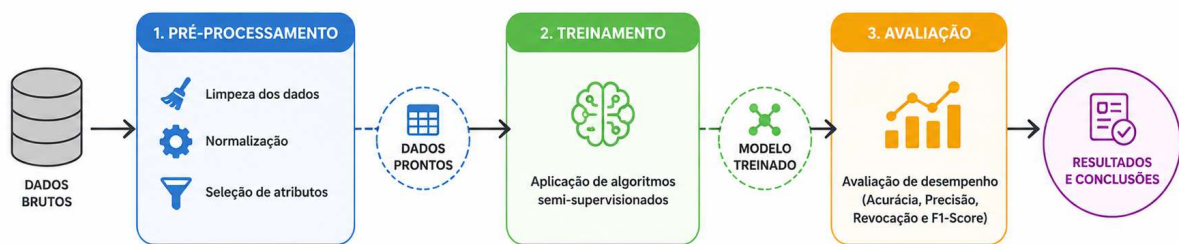


Figura 1 – Fluxo geral do processo de classificação em aprendizado de máquina

Fonte: Elaborado pelo autor (2026).

2.2.1 PRÉ-PROCESSAMENTO

O pré-processamento corresponde ao conjunto de técnicas aplicadas aos dados brutos com o objetivo de torná-los adequados para a aplicação de algoritmos de aprendizado de máquina (Géron, 2022). No contexto da detecção de intrusão em redes de computadores, essa etapa é relevante devido à alta dimensionalidade do tráfego, à presença de ruídos, valores inconsistentes e ao forte desbalanceamento entre classes Sommer e Paxson (2010).

Em cenários que utilizam aprendizado semi-supervisionado, o pré-processamento assume papel ainda mais crítico, uma vez que a qualidade da representação dos dados influencia a propagação de rótulos e a identificação de similaridades entre instâncias rotuladas e não rotuladas Engelen e Hoos (2020). Dessa forma, um pré-processamento adequado é essencial para permitir que os modelos explorem de forma eficiente as estruturas presentes nos dados de tráfego de rede (CHAPELLE; SCHÖLKOPF; ZIEN, 2006).

Entre as principais atividades de pré-processamento destacam-se a limpeza dos dados, tratamento de valores ausentes, normalização de atributos e seleção de características relevantes. Essas técnicas contribuem para reduzir a complexidade dos dados e melhorar a capacidade de generalização dos modelos, impactando diretamente o desempenho dos classificadores utilizados em sistemas de detecção de intrusão.

A normalização merece destaque nesse conjunto, pois atributos de tráfego de rede apresentam escalas distintas entre si, como durações de fluxo na ordem de milhões e contagens de flags em valores próximos de zero. Sem um ajuste de escala, atributos de maior magnitude tendem a dominar o cálculo de distâncias enviesando o modelo. Uma das técnicas mais empregadas para mitigar esse problema é a padronização *z-score*, que transforma cada atributo para que possua média zero e desvio-padrão unitário, conforme a Equação 2.1:

$$z = \frac{x - \mu}{\sigma} \quad (2.1)$$

onde x é o valor original do atributo, μ a média e σ o desvio-padrão calculados sobre o conjunto de treino. Dessa forma, todos os atributos passam a compartilhar uma mesma escala de referência, evitando que diferenças de magnitude enviesem o modelo.

2.2.2 TREINAMENTO

A etapa de treinamento corresponde ao processo de construção dos modelos de aprendizado de máquina a partir dos dados previamente pré-processados, sendo responsável por ajustar os parâmetros internos dos algoritmos de forma a representar padrões presentes no tráfego de rede. No contexto da detecção de intrusão, essa fase é particularmente desafiadora devido à escassez de dados rotulados, ao desbalanceamento entre classes e à presença de ataques raros ou desconhecidos [Zheng, Yang e Wang \(2023b\)](#).

Nas abordagens convencionais encontradas na literatura, grande parte dos Sistemas de Detecção de Intrusão (IDS) é baseada em paradigmas de aprendizado supervisionado. Nesse tipo de treinamento, o algoritmo aprende a mapear entradas específicas a partir de dados previamente rotulados ([ALPAYDIN, 2020](#)). Embora esses modelos frequentemente apresentem alto desempenho na classificação de amostras conhecidas, quando aplicados em ambientes reais podem enfrentar dificuldades para se adaptar a novos padrões de ataque, sendo necessário uma rica diversidade de dados rótulados afim de esturdar o comportamento das amostras. Como consequência, tendem a apresentar limitações na detecção de ameaças inéditas, com os chamados *zero-day attacks* ([MVULA et al., 2024](#)).

No escopo deste trabalho, o treinamento é fundamentado em modelos de aprendizado semi-supervisionado. Essas abordagens combinam um pequeno conjunto de instâncias rotuladas com grandes volumes de dados não rotulados, explorando a estrutura intrínseca

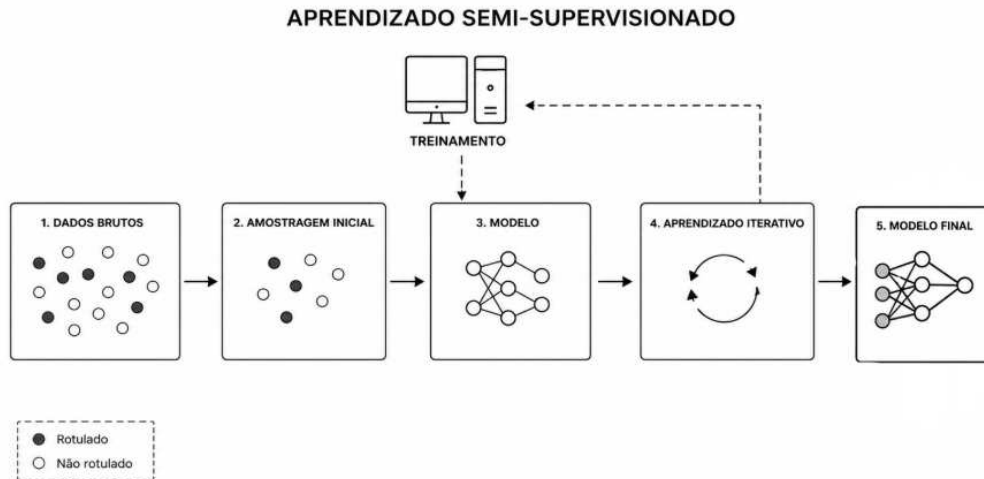


Figura 2 – Fluxo Semi-supervisionado

Fonte: Elaborado pelo autor (2026).

dos dados para orientar o processo de aprendizagem [Zhu \(2009\)](#). Entre os modelos mais utilizados destacam-se métodos baseados em propagação de rótulos, como *Label Propagation*, e estratégias iterativas de auto-rotulagem, como o *Self-Training*, além de técnicas que utilizam medidas de similaridade ou fronteiras de decisão apoiadas por dados não rotulados [Xiang, Yong e Meng \(2010\)](#).

Em IDS, durante o treinamento de propostas semi-supervisionadas, as instâncias rotuladas fornecem informação inicial sobre o comportamento normal ou malicioso, enquanto as instâncias não rotuladas contribuem para capturar a distribuição global do tráfego de rede ([CHAPELLE; SCHÖLKOPF; ZIEN, 2006](#)). Essa estratégia permite ampliar a capacidade de generalização dos modelos e torna o processo mais adequado a cenários reais, nos quais a rotulagem manual é limitada ou inviável. Dessa forma, o treinamento semi-supervisionado estabelece a base conceitual para a etapa de avaliação dos métodos de detecção de intrusão.

Conforme ilustrado na Figura 2, o aprendizado semi-supervisionado inicia-se a partir de um conjunto de dados brutos, composto majoritariamente por instâncias não rotuladas e complementado por um pequeno subconjunto rotulado. A partir dessas informações iniciais, o modelo é treinado utilizando os exemplos rotulados como referência, enquanto explora a estrutura dos dados não rotulados para inferir padrões adicionais. Durante esse processo, novas amostras podem ser incorporadas ao treinamento por meio de mecanismos de realimentação, nos quais rótulos estimados pelo próprio modelo são gradualmente utilizados para refinar o aprendizado. O algoritmo, ao integrar essas diferentes fontes de informação, produz um modelo treinado capaz de generalizar melhor o comportamento do tráfego de rede, tornando-se mais robusto à presença de ataques desconhecidos e variações

nos dados observados.

2.2.3 AVALIAÇÃO

A etapa de avaliação corresponde à análise do desempenho dos modelos de detecção de intrusão obtidos durante o treinamento, sendo fundamental independentemente da técnica de aprendizado adotada (WITTEN et al., 2016). No contexto da segurança em redes de computadores, a avaliação permite verificar a capacidade do modelo em distinguir corretamente o tráfego legítimo de atividades maliciosas, além de possibilitar a análise dos impactos causados por erros de classificação em ambientes sensíveis à segurança Williams e Qian (2025).

Em problemas de classificação binária, amplamente utilizados em sistemas de detecção de intrusão, as predições realizadas pelo modelo podem ser categorizadas em quatro tipos: verdadeiros positivos (*True Positive* – TP), verdadeiros negativos (*True Negative* – TN), falsos positivos (*False Positive* – FP) e falsos negativos (*False Negative* – FN). De modo geral, a classe negativa está associada ao tráfego normal da rede, enquanto a classe positiva representa comportamentos anômalos ou ataques. Os termos “verdadeiro” e “falso” indicam se a classe atribuída pelo modelo corresponde corretamente ao rótulo real da instância avaliada.

A partir dessas categorias, diferentes métricas podem ser utilizadas para mensurar o desempenho dos modelos. Uma das mais empregadas é a **acurácia**, que mede a proporção de classificações corretas em relação ao total de instâncias avaliadas, conforme a Equação 2.2.

$$\text{Acurácia} = \frac{TP + TN}{TP + FP + TN + FN} \quad (2.2)$$

Porém, vale ressaltar que, em cenários desbalanceados, tal métrica por si só nem sempre é um bom indicador, uma vez que ela não identifica a desproporção de acertos, sendo necessário o apoio das métricas a seguir.

A precisão avalia a assertividade das previsões positivas realizadas pelo modelo, indicando a proporção de instâncias corretamente classificadas como ataque em relação ao total de instâncias classificadas como positivas, conforme a Equação 2.3.

$$\text{Precisão} = \frac{TP}{TP + FP} \quad (2.3)$$

A revocação (*recall*) mede a capacidade do modelo em identificar corretamente todas as instâncias pertencentes à classe positiva, sendo definida conforme a Equação 2.4.

$$\text{Revocação} = \frac{TP}{TP + FN} \quad (2.4)$$

Por fim, o F1-Score combina precisão e revocação em um único valor por meio da média harmônica dessas duas métricas. Essa medida é particularmente relevante em cenários de detecção de intrusão, nos quais há forte desbalanceamento entre classes e erros de classificação podem gerar impactos significativos, conforme a Equação 2.5.

$$F1-Score = 2 \cdot \frac{\text{Precisão} \cdot \text{Revocação}}{\text{Precisão} + \text{Revocação}} \quad (2.5)$$

Para garantir uma avaliação coerente e com alta capacidade de generalização, não basta apenas empregar métricas adequadas, é imprescindível assegurar uma amostragem representativa dos dados. Antes de realizar o treinamento dos modelos, há a necessidade de gerar um ambiente que reflita a complexidade do tráfego real. Isso exige uma separação estratificada, garantindo bases de treino e teste que consigam atingir e representar todas as vertentes de agrupamentos possíveis, abrangendo comportamentos legítimos e ameaças.

Sendo assim, técnicas clássicas de particionamento de dados, como o *Holdout*, são aplicadas na etapa de preparação das amostras para garantir essa estabilidade e representatividade (SOMMER; PAXSON, 2010). O método *Holdout* destaca-se pela sua simplicidade e agilidade computacional, baseando-se na divisão do conjunto de dados em subconjuntos distintos e mutuamente exclusivos, destinados ao treinamento e ao teste. Por ser uma abordagem flexível, o *Holdout* engloba diversas variações de amostragem, desde a partição puramente aleatória (*Standard Holdout*), passando pela manutenção rigorosa das proporções originais das classes (*Stratified Holdout*). Essa versatilidade o torna um excelente candidato para a avaliação de modelos semi-supervisionados no contexto da detecção de intrusão em redes computacionais.

2.3 Algoritmos Semi-supervisionados Clássicos

Entre os algoritmos de aprendizado semi-supervisionado mais amplamente citados na literatura destacam-se o *Label Propagation* e o *Self-Training*. Esses métodos são frequentemente adotados em diferentes domínios devido à sua simplicidade conceitual e à capacidade de explorar grandes volumes de dados não rotulados.

O algoritmo *Label Propagation* baseia-se na construção de um grafo de similaridade, no qual cada instância é representada como um nó conectado a outros nós conforme uma medida de proximidade. A partir desse grafo, os rótulos conhecidos são propagados para as instâncias não rotuladas de forma iterativa, assumindo que dados semelhantes tendem a pertencer à mesma classe. Essa abordagem é particularmente adequada a problemas de detecção de intrusão, nos quais padrões de tráfego normal e malicioso apresentam estruturas recorrentes Chapelle, Schölkopf e Zien (2006).

Já o *Self-Training* consiste em uma estratégia iterativa na qual um classificador

supervisionado é inicialmente treinado com um conjunto reduzido de dados rotulados. Em seguida, o modelo passa a rotular automaticamente as instâncias não rotuladas nas quais apresenta maior confiança, incorporando-as progressivamente ao conjunto de treinamento. Esse processo permite ampliar a base de dados rotulados ao longo do tempo, reduzindo a dependência de rotulagem manual e tornando o modelo mais adaptável a cenários dinâmicos [Zhu \(2009\)](#).

Uma extensão natural do *Self-Training* é o *Co-Training*. Nessa estratégia, em vez de um único classificador, dois modelos são treinados simultaneamente, onde a cada iteração o classificador rotula as instâncias não rotuladas, que apresenta maior confiança e as enriquece fornecendo ao outro modelo, que as incorpora ao seu conjunto de treinamento. Essa troca colaborativa permite que os dois classificadores se reforcem mutuamente, ampliando o aproveitamento dos dados não rotulados ([BLUM; MITCHELL, 1998](#)). O *Co-Training* pode ser combinado com diferentes classificadores supervisionados como base, quando associado ao *Random Forest*, por exemplo, cada modelo aprende por um conjunto de árvores de decisão, estratégia adotada em trabalhos recentes de detecção de intrusão [Williams e Qian \(2025\)](#).

2.4 Trabalhos Relacionados

Abordando a dificuldade e o alto custo associados à coleta e simulação de amostras de tráfego malicioso, o trabalho de ([SVIHROVÁ; LETTNER, 2020](#)) propõe uma abordagem de detecção de intrusão baseada em aprendizado semi-supervisionado. O estudo utiliza o dataset NSL-KDD, derivado do KDD 99, contendo 41 atributos de conexões de rede. Durante o pré-processamento, variáveis categóricas foram convertidas por meio de codificação one-hot, enquanto atributos contínuos foram normalizados utilizando a transformação min-max. O método central baseia-se em um autoencoder treinado exclusivamente com amostras benignas. Dessa forma, amostras anômalas produzem erros de reconstrução significativamente maiores. A classificação final é realizada utilizando o erro de reconstrução como métrica, combinado a um modelo de mistura Gaussiana para definição automática do limiar de decisão, alcançando 89,71% de acurácia no conjunto de testes. Entretanto, observa-se que a abordagem empregada se aproxima mais de um método de detecção de anomalias baseado em *one-class classification*, uma vez que o treinamento é realizado apenas com amostras benignas, sem a utilização conjunta de dados rotulados e não rotulados. Dessa forma, apesar da relevância da metodologia e das contribuições relacionadas ao pré-processamento e ao uso de autoencoders, o modelo não se enquadra estritamente no paradigma de aprendizado semi-supervisionado.

Pensando sob a mesma óptica, não é apenas difícil lidar com a escassez de rótulos malignos, mas também manipular grandes volumes de dados rotulados em ambientes de rede.

Nesse contexto, (WILLIAMS; QIAN, 2025) investigam o uso de técnicas de aprendizado semi-supervisionado para detecção de intrusões em redes de grande escala. O trabalho propõe três estratégias principais, que foram antecipadas pelo pré-processamento, sendo eles, o tratamento de valores ausentes e inconsistentes, remoção de registros redundantes e transformação de atributos categóricos por meio de técnicas como label encoding e feature hashing. A primeira utiliza Decision Tree com amostragem de incerteza baseada em entropia, selecionando instâncias não rotuladas mais incertas para rotulação iterativa (active learning). A segunda emprega Regressão Logística com self-training, na qual o modelo é inicialmente treinado em um conjunto de dados rotulado e, posteriormente, utilizado para prever rótulos em dados não rotulados. Em seguida, as instâncias classificadas com maior confiança são adicionadas ao conjunto rotulado, permitindo o re-treinamento do modelo. Por fim, a terceira abordagem utiliza co-training com Random Forest, em que dois classificadores treinados em subconjuntos distintos de atributos rotulam colaborativamente instâncias não rotuladas; aquelas com maior nível de confiança são então utilizadas para re-treinar o outro classificador. Os experimentos, conduzidos com o dataset CIC-DDoS2019, avaliaram cenários com 1% a 50% de dados rotulados, alcançando resultados próximos de 99% de acurácia em alguns modelos.

Seguindo a mesma perspectiva da exploração de dados semi-supervisionados, utilizando técnicas de distribuição de rótulos entre bases de dados, (SARANTOS; VIOLOS; LEI-VADEAS, 2025) apresentam uma abordagem que combina métodos semi-supervisionados e supervisionados para melhorar a generalização dos modelos. No trabalho, o algoritmo Label Propagation é utilizado para propagar rótulos para dados originalmente não rotulados, permitindo ampliar o conjunto de treinamento. Em seguida, os modelos supervisionados Decision Tree e Random Forest são treinados utilizando esse conjunto enriquecido. Inicialmente, os segmentos de dados foram compostos por 80% de registros rotulados do dataset CIC-IDS2017 e 20% de registros não rotulados do PS-Azure2023. Além disso, na parte de pré-processamento foi utilizado a ferramenta CICFlowMeter e aplicada a técnica de Information Gain para seleção de atributos, reduzindo a dimensionalidade e mantendo apenas as características mais relevantes. Os resultados mostram acurácia superior a 99% quando treinamento e teste ocorrem no mesmo dataset, mas com queda para cerca de 79% a 84% quando avaliados em ambientes diferentes.

Olhando sob a ótica de valorização das informações contidas no dataset, o trabalho de (YUAN et al., 2019) propõe o algoritmo *Semi-Supervised Tri-Adaboost* (STA) para detecção de intrusões em redes. O estudo utiliza o dataset KDDcup99, amplamente empregado como benchmark em sistemas de detecção de intrusão. Durante o pré-processamento, foi aplicada a técnica de seleção de atributos baseada no método Qui-Quadrado, responsável por reduzir a dimensionalidade do conjunto de dados e selecionar as features mais relevantes para o processo de classificação, evitando a “Maldição da dimensionalidade”. A abordagem combina três variações do algoritmo Adaboost — *Discrete*, *Real* e *Gentle* —

integradas por meio do método de *tri-training*, no qual três classificadores colaboram para rotular iterativamente amostras não rotuladas com base no consenso entre os modelos. Os experimentos demonstram que o método consegue alcançar aproximadamente 98,08% de precisão utilizando apenas uma pequena fração de dados rotulados, evidenciando a eficiência da abordagem semi-supervisionada.

2.4.1 TABELA COMPARATIVA DOS TRABALHOS RELACIONADOS

Após a leitura dos artigos relacionados, foram selecionados aqueles considerados mais relevantes, citados anteriormente, os quais foram encontrados por meio da Biblioteca da Universidade Federal de Uberlândia (UFU), de ferramentas de busca científica, como o Google Scholar, bem como por meio de indicações de leitura.

A Tabela 1 sintetiza as principais características dos trabalhos relacionados identificados, facilitando a visualização das semelhanças, diferenças e lacunas que motivam a presente pesquisa.

Tabela 1 – Comparação entre abordagens semi-supervisionadas para detecção de intrusão

Trabalho	Base de Dados	Pré-processamento	Algoritmos / Classificadores	Resultados
(SVIHROVÁ; LETTNER, 2020)	NSL-KDD	Codificação <i>One-Hot</i> para variáveis categóricas e normalização <i>Min-Max</i> para atributos contínuos	<i>Autoencoder</i> treinado com amostras benignas + Modelo de Mistura Gaussiana para definição do limiar de decisão	<i>Recall</i> : 92,38%
(WILLIAMS; QIAN, 2025)	CIC-DDoS2019	Limpeza dos Dados, redução de redundância nos registros, <i>label encoding</i> e <i>feature hashing</i> .	<i>Decision Tree</i> com <i>Active Learning</i> , Regressão Logística com <i>Self-Training</i> e <i>Random Forest</i> com <i>Co-Training</i>	<i>Recall</i> 85,46% (Random Forest) e <i>Recall</i> 84,71% (<i>Decision Tree</i>)
(SARANTOS; VIOLOS; LEIVADEAS, 2025)	CIC-IDS2017 + PS-Azure2023	Extração de características com <i>CIC-FlowMeter</i> e seleção de atributos utilizando Ganho de Informação	<i>Label Propagation</i> para rotulação de dados não rotulados + <i>Decision Tree</i> e <i>Random Forest</i>	<i>Recall</i> : 99.89%
(YUAN et al., 2019)	KDDCup99	Seleção de atributos baseada no método Qui-Quadrado	Semi-Supervised Tri-AdaBoost (STA) combinando AdaBoost (Discrete, Real e Gentle) com Tri-Training	Semi-supervised tri-Adaobst (STA) : 89,05% <i>Recall</i>

2.5 Considerações Finais do Capítulo

Este capítulo apresentou os principais conceitos teóricos relacionados à detecção de intrusão em redes de computadores, com ênfase na aplicação de modelos de aprendizado semi-supervisionado. Foram discutidos os fundamentos dos sistemas de detecção de intrusão, os processos de pré-processamento, treinamento e avaliação, bem como os algoritmos semi-supervisionados clássicos utilizados nesse contexto.

Além disso, foram discutidas as métricas de desempenho e os métodos de separação dos dados de treinamento e teste. Nesse contexto, explorou-se o método Holdout como técnica de particionamento para avaliação dos modelos. Por fim, a fundamentação teórica apresentada fornece o embasamento conceitual e prático necessário para realizar uma amostragem adequada, garantindo a formação de uma base de dados representativa e com elevada capacidade de generalização, contribuindo para o desenvolvimento de modelos mais precisos.

3 MÉTODO PARA COMPARAÇÃO DE ALGORITMOS SEMI-SUPERVISIONADOS PARA DETECÇÃO DE INTRUSÃO

O objetivo deste capítulo é apresentar o método utilizado para comparar diferentes algoritmos de aprendizado semi-supervisionado e seu impacto preditivo na tarefa de detecção de intrusão em redes de computadores. A seguir 3.1 apresenta uma visão geral do método proposto. A Seção 3.2 detalha a aquisição e organização da base de dados utilizada nos experimentos. A Seção 3.3 descreve o pré-processamento aplicado aos dados. Em seguida, seção 3.4 apresenta os dois algoritmos semi-supervisionados investigados. A Seção 3.5 discorre sobre as condições de treinamento dos modelos, incluindo as frações progressivas de rotulagem empregadas e o tratamento do desbalanceamento com a técnica SMOTE. Por fim, a Seção 3.6 apresenta as métricas utilizadas na avaliação e validação dos resultados obtidos.

3.1 Visão Geral do Método

A Figura 3 apresenta uma visão geral do método experimental proposto, organizado em quatro etapas encadeadas. Inicialmente, a base de dados passa pelo estágio de pré-processamento, no qual os dados de tráfego da base são limpos, normalizados e submetidos à seleção de atributos (Seção 3.3). Em seguida, o conjunto resultante é dividido por particionamento *Holdout* em treino e teste, e frações progressivas de rótulos sobre o treino são selecionadas para simular a escassez de dados rotulados. Cabe destacar que as bases selecionadas possuem desbalanceamento natural entre classes e tráfego normal e de ataque, característica inerente ao cenário real. O desbalanceamento será tratado na Seção 3.5, por meio da técnica SMOTE, visando melhorar a representatividade nas amostras em cenários de alto desbalanceamento.

A partir dessas frações, os dois algoritmos investigados, o *Label Propagation* e o *Random Forest* com *Co-Training*, são treinados sob condições experimentais idênticas, de modo que eventuais diferenças de desempenho decorram do algoritmo, e não da partição dos dados (Seção 3.4). Por fim, as predições sobre o conjunto de teste são confrontadas com os rótulos verdadeiros, e o desempenho é mensurado por métricas extraídas da matriz de confusão (Seção 3.6).

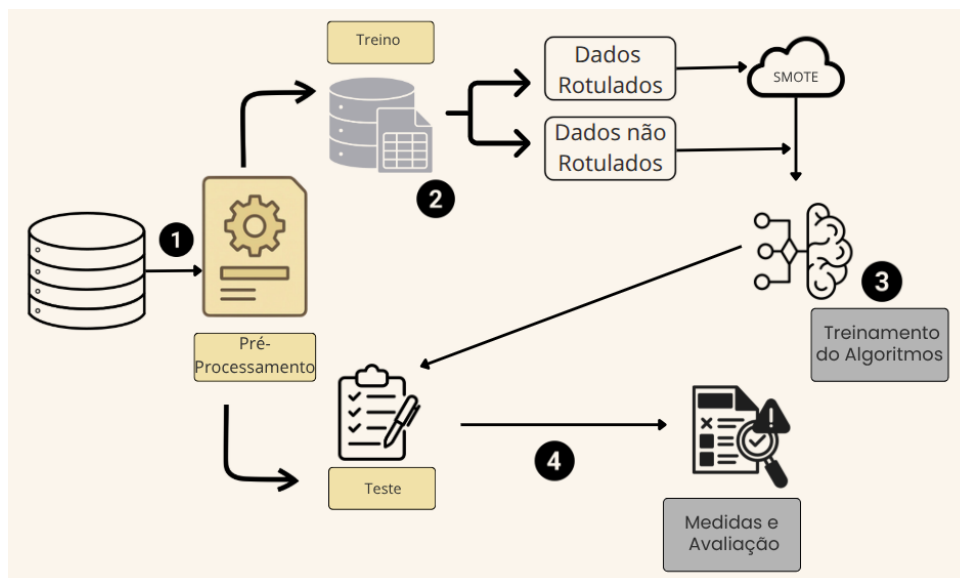


Figura 3 – Fluxo geral do método proposto para comparar algoritmos semi-supervisionados.

Fonte: Elaborado pelo autor (2026).

3.2 Aquisição e Organização dos Dados

A utilização de bases de dados representativas é fator essencial no desenvolvimento e na avaliação de sistemas de detecção de intrusão em redes de computadores. Em cenários reais, o tráfego de rede apresenta grande diversidade de comportamentos, abrangendo tanto comunicações legítimas quanto diferentes tipos de atividades maliciosas. Dessa forma, o emprego de conjuntos de dados confiáveis, que representem adequadamente essas condições, torna-se fundamental para garantir que os modelos desenvolvidos identifiquem padrões de ataque de maneira eficaz e generalizável.

Nesse contexto, para a realização dos experimentos foi selecionado o conjunto de dados *Intrusion Detection Evaluation Dataset* (CIC-IDS2017), desenvolvido pelo *Canadian Institute for Cybersecurity* (CIC). A base foi coletada entre os dias 3 e 7 de julho de 2017, em um ambiente de rede controlado no qual diferentes tipos de ataques foram executados juntamente com atividades legítimas de usuários. Os autores classificaram os ataques em seis perfis principais: força bruta, *Heartbleed*, negação de serviço (DoS), negação de serviço distribuída (DDoS), *Botnet*, ataques a aplicações web e infiltração. Essa estratégia permitiu gerar um conjunto contendo tráfego normal e tráfego malicioso devidamente rotulado, viabilizando seu uso em tarefas de aprendizado semi-supervisionado.

Outro aspecto relevante da CIC-IDS2017 é a organização dos ataques ao longo dos dias de captura. Cada dia contempla diferentes cenários de ataque e diferentes proporções entre tráfego legítimo e malicioso, o que permite avaliar os modelos em múltiplos contextos de rede. Durante o estudo da base de dados foi identificado dias com volumes de rótulo

e características de ataques diferentes, conforme apresentado na Tabela 2. Dessa forma, visando analisar o comportamento dos modelos em diferentes cenários de treinamento, optou-se pela utilização da base de terça-feira, devido ao maior desbalanceamento e à presença de um tipo específico de ataque, e da base de sexta-feira, por apresentar uma distribuição mais equilibrada e contemplar múltiplos tipos de ataques. As amostras possuem uma coluna de rotulagem denominada Rótulo (*Label*), responsável por indicar se o fluxo corresponde a tráfego benigno ou a um ataque, sendo utilizada como variável alvo no treinamento e na avaliação dos modelos. A Tabela 3 apresenta a relação dos 74 atributos presentes na base, incluindo o próprio rótulo, que ocupa a última posição do conjunto.

Tabela 2 – Distribuição dos ataques por dia da semana e período do dia.

Dia da Semana	Período	Ataque	Distribuição
Terça-feira	Manhã	Força Bruta	Desbalanceada
	Tarde	Força Bruta	Desbalanceada
Quarta-feira	Manhã	DoS	Balanceada
	Tarde	Heartbleed	Balanceada
Quinta-feira	Manhã	Web	Desbalanceada
	Tarde	Infiltração	Desbalanceada
Sexta-feira	Manhã	Botnet	Balanceada
	Tarde	DDoS e Escaneamento de Portas1	Balanceada

Fonte: Canadian Institute for Cybersecurity, CIC-IDS2017.

Fonte: Elaborado pelo autor com base na documentação do CIC-IDS2017.

3.3 Pré-processamento dos Dados

Antes de serem submetidos aos modelos de aprendizado, os dados passaram por uma etapa de preparação destinada a corrigir inconsistências e adequar os atributos ao formato esperado pelos algoritmos. O mesmo procedimento foi aplicado de forma independente às bases de terça-feira e de sexta-feira.

Inicialmente, utilizando a biblioteca **pandas** do Python, realizou-se a limpeza dos dados. Os valores infinitos (**np.inf** e **-np.inf**), comuns em atributos de taxa, como **Flow Bytes/s** e **Flow Packets/s**, foram convertidos em valores ausentes (**NaN**) por meio do método **replace**. Em seguida, todos os registros contendo valores ausentes foram removidos com o método **dropna** do Python. Na base de terça-feira, esse processo reduziu o conjunto de 445.909 para 445.645 registros; na de sexta-feira, de 703.245 para 702.718 registros.

Em seguida, a variável alvo (*Label*) foi separada dos atributos preditores e convertida para o tipo inteiro utilizando o método **astype(int)** do **pandas**, representando o tráfego legítimo como 0 e as amostras de ataque como 1. Os atributos numéricos restantes foram normalizados utilizando o método **StandardScaler** da biblioteca **scikit-learn**. Por fim,

Tabela 3 – Atributos presentes na base de dados CIC-IDS2017.

Nº	Atributo	Nº	Atributo	Nº	Atributo
1	Flow Duration	26	Fwd PSH Flags	51	Fwd Avg Packets/Bulk
2	Fwd Packet Length Max	27	Bwd PSH Flags	52	Fwd Avg Bulk Rate
3	Fwd Packet Length Min	28	Fwd URG Flags	53	Bwd Avg Bytes/Bulk
4	Fwd Packet Length Mean	29	Bwd URG Flags	54	Bwd Avg Packets/Bulk
5	Fwd Packet Length Std	30	Fwd Header Length	55	Bwd Avg Bulk Rate
6	Bwd Packet Length Max	31	Bwd Header Length	56	Subflow Fwd Packets
7	Bwd Packet Length Min	32	Fwd Packets/s	57	Subflow Fwd Bytes
8	Bwd Packet Length Mean	33	Bwd Packets/s	58	Subflow Bwd Packets
9	Bwd Packet Length Std	34	Min Packet Length	59	Subflow Bwd Bytes
10	Flow Bytes/s	35	Max Packet Length	60	Init_Win_bytes_forward
11	Flow Packets/s	36	Packet Length Mean	61	Init_Win_bytes_backward
12	Flow IAT Mean	37	Packet Length Std	62	act_data_pkt_fwd
13	Flow IAT Std	38	Packet Length Variance	63	min_seg_size_forward
14	Flow IAT Max	39	FIN Flag Count	64	Active Mean
15	Flow IAT Min	40	SYN Flag Count	65	Active Std
16	Fwd IAT Total	41	RST Flag Count	66	Active Max
17	Fwd IAT Mean	42	PSH Flag Count	67	Active Min
18	Fwd IAT Std	43	ACK Flag Count	68	Idle Mean
19	Fwd IAT Max	44	URG Flag Count	69	Idle Std
20	Fwd IAT Min	45	ECE Flag Count	70	Idle Max
21	Bwd IAT Total	46	Down/Up Ratio	71	Idle Min
22	Bwd IAT Mean	47	Average Packet Size	72	Sum Fwd Packets
23	Bwd IAT Std	48	Avg Fwd Segment Size	73	Sum Length of Fwd Packets
24	Bwd IAT Max	49	Avg Bwd Segment Size	74	Label
25	Bwd IAT Min	50	Fwd Avg Bytes/Bulk		

Fonte: Elaborado pelo autor com base na documentação da base CIC.

com o objetivo de reduzir a dimensionalidade do problema, aplicou-se a seleção de atributos descrita na Subseção 3.3.1.

3.3.1 SELEÇÃO DE ATRIBUTOS

Dada a alta dimensionalidade do conjunto, aplicou-se uma etapa de seleção de atributos baseada no teste estatístico Qui-Quadrado ($\chi^2 = \sum \frac{(O_i - E_i)^2}{E_i}$), que avalia o grau de dependência entre cada atributo preditor e a classe alvo binarizada: quanto maior o valor de χ^2 e menor o respectivo p -valor, maior a relevância do atributo para distinguir tráfego benigno de malicioso. Com base na abordagem apresentada por (YUAN et al., 2019), essa técnica é eficaz em cenários com alta dimensionalidade, pois ajuda a reduzir os impactos da chamada maldição da dimensionalidade. Esse fenômeno ocorre quando um conjunto de dados possui um grande número de variáveis (dimensões), fazendo com que os dados fiquem cada vez mais dispersos no espaço e como consequência, podem resultar em perda de desempenho.

O teste identificou nove atributos com relevância nula em ambas as bases, por se tratarem de colunas constantes (valor único em todos os registros), o que impossibilita

a medição de dependência com a classe alvo. Esses atributos, apresentados na Tabela 4, foram descartados, reduzindo o conjunto para 64 colunas em ambas as bases.

Tabela 4 – Atributos descartados na seleção de características por apresentarem relevância nula.

Atributo	Motivo da remoção
Bwd PSH Flags	Variância nula
Fwd URG Flags	Variância nula
Bwd URG Flags	Variância nula
Fwd Avg Bytes/Bulk	Variância nula
Fwd Avg Packets/Bulk	Variância nula
Fwd Avg Bulk Rate	Variância nula
Bwd Avg Bytes/Bulk	Variância nula
Bwd Avg Packets/Bulk	Variância nula
Bwd Avg Bulk Rate	Variância nula

Fonte: Elaborado pelo autor.

Observou-se ainda que a ordem de relevância dos demais atributos variou conforme a natureza do ataque em cada base. Na de terça-feira, despontaram características associadas ao comprimento dos pacotes encaminhados (`Fwd Packet Length Std`) e a *flags* de sinalização (`Fwd PSH Flags`, `SYN Flag Count`); na de sexta-feira, predominante em tráfego DDoS, os maiores escores couberam a `PSH Flag Count` e `URG Flag Count`. Essa diferença reforça a importância de avaliar a relevância dos atributos em cada cenário, em vez de assumir um conjunto fixo de características como universalmente discriminativo.

3.4 Algoritmos Semi-Supervisionados

Existem diversas abordagens de aprendizado semi-supervisionado propostas na literatura, cada qual com mecânicas distintas para explorar dados não rotulados, escolhidas conforme a natureza dos dados e os objetivos da análise. Dentre elas, foram selecionados, por sua forte presença na literatura e pela aderência direta aos trabalhos de referência adotados, os seguintes algoritmos:

- *Label Propagation Algorithm* (LPA) ([SARANTOS; VIOLOS; LEIVADEAS, 2025](#));
- *Random Forest* com *Co-Training* ([WILLIAMS; QIAN, 2025](#)).

A implementação foi conduzida na linguagem Python, amplamente adotada entre cientistas de dados por sua natureza interativa e pelo ecossistema crescente de bibliotecas científicas. Empregaram-se, principalmente, as bibliotecas *Pandas*, *NumPy* e *Scikit-learn*, esta última responsável por implementações consolidadas de diversos algoritmos de aprendizado de máquina, mantendo uma interface coesa e integrada à linguagem [Pedregosa et al.](#)

(2011). O LPA foi utilizado diretamente a partir da classe `LabelPropagation` do módulo `sklearn.semi_supervised`, enquanto o *Co-Training* foi implementado de forma própria sobre classificadores `RandomForestClassifier` do módulo `sklearn.ensemble`, já que o *framework* não disponibiliza rotina nativa para essa estratégia. Sendo assim, o código-fonte dos algoritmos utilizados nos experimentos está disponível no [repositório](#) do projeto.

O Algoritmo 1 sintetiza o fluxo do *Label Propagation Algorithm*, configuração adotada por Sarantos (SARANTOS; VIOLOS; LEIVADEAS, 2025), usando parâmetros do Kernel igual a Knn, neighbors igual a 7, e TOL igual a 1e-3. A técnica parte de uma ideia simples: instâncias parecidas tendem a pertencer à mesma classe. Para explorar isso, o algoritmo monta um grafo em que cada instância, rotulada ou não, é um nó, ligado aos seus k vizinhos mais próximos (Algoritmo 1, linha 1). Os poucos nós que já possuem rótulo funcionam como âncoras (Algoritmo 1, linha 2) e espalham seu rótulo pelo grafo em iterações sucessivas (Algoritmo 1, linha 4), enquanto os nós originalmente rotulados são restaurados ao seu valor verdadeiro a cada passo, funcionando como âncoras fixas (Algoritmo 1, linha 5). O processo se encerra quando os rótulos se estabilizam ou o número máximo de iterações é atingido (Algoritmo 1, linha 6). Ao final, cada instância não rotulada recebe a classe predominante em seu nó Algoritmo (1, linha 7).

Algorithm 1 Fluxo do *Label Propagation Algorithm* (LPA) com *kernel* KNN

Require: conjunto rotulado L ; conjunto não rotulado U ; número de vizinhos k ; máximo de iterações

Ensure: rótulos preditos $\hat{y}$ para as instâncias de U

- 1: Construir um grafo unindo cada instância de $L \cup U$ aos seus k vizinhos mais próximos.
- 2: Atribuir a cada nó de L o seu rótulo verdadeiro; deixar os nós de U sem rótulo.
- 3: **repeat**
- 4: Cada nó atualiza seu rótulo com base nos rótulos de seus vizinhos.
- 5: Restaurar os nós de L ao seu rótulo verdadeiro (âncoras fixas).
- 6: **until** os rótulos se estabilizam ou o máximo de iterações é atingido
- 7: Atribuir a cada instância de U a classe predominante no nó final.
- 8: **return** $\hat{y}$

Fonte: Adaptado de Zhu (ZHENG; YANG; WANG, 2023b).

O Algoritmo 2, juntamente com a Figura 4, sintetizam o fluxo do algoritmo de *Co-training* adotado, traduzido do pseudocódigo de Williams e Qian (WILLIAMS; QIAN, 2025), os quais utilizam os parâmetros *randomState* igual a 42, número de instâncias confiantes igual a 100, número máximo de iterações igual a 10 e, por fim, limiar de confiança igual a 0,9. Inicialmente, o conjunto rotulado é duplicado para formar os dois conjuntos de treinamento (Algoritmo 2, linha 2), enquanto os atributos são divididos parcialmente para cada modelo (Algoritmo 2, linha 1). Cada *Random Forest* é treinado utilizando apenas o seu subconjunto de atributos (Algoritmo, 2, linha 3).

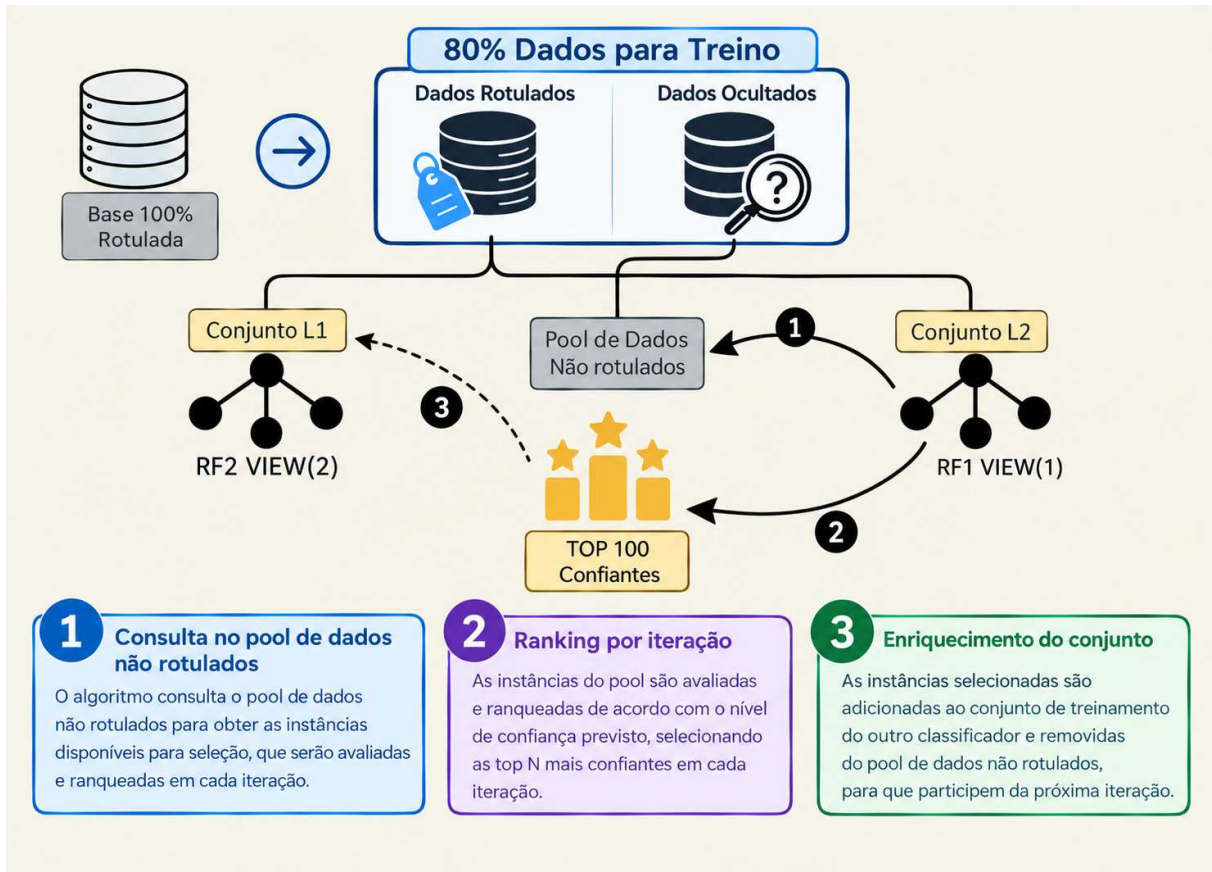


Figura 4 – Fluxo do Co-Training.

Fonte: Elaborado pelo autor (2026).

3.5 Treinamento dos Modelos Semi-Supervisionados

Em seguida, ambos os classificadores realizam predições sobre o conjunto de dados não rotulados (Algoritmo 2, linha 8). As instâncias classificadas com maior confiança, delimitadas pela quantidade informada (n), são selecionadas (Algoritmo 2, linha 9) e adicionadas ao conjunto de treinamento do outro classificador (Algoritmo 2, linha 13). Após essa troca de informações, as instâncias incorporadas são removidas do conjunto não rotulado (linha 15 2).

Esse processo é repetido até que não existam mais instâncias no conjunto de não rotulados ou seja atingido o número máximo de iterações (Algoritmo 2, linha 16). Ao final, são obtidos dois classificadores treinados de forma colaborativa (Algoritmo 2, linha 17).

Por fim, após o treinamento dos dois modelos Random Forest, é realizada a etapa de classificação do conjunto de teste para avaliar o desempenho da abordagem proposta (Algoritmo 2, linha 18). Nessa fase, ambos os classificadores percorrem as instâncias do conjunto de teste e realizam suas respectivas predições com base no conhecimento adquirido durante o treinamento. Para cada amostra, será imposto um nível de confiança ao rotulado

julgado (Algoritmo 2, linha 20), sendo adotada como predição final aquela produzida pelo classificador com maior confiança (Algoritmo 2, linha 21). Em situações de empate, a decisão é atribuída ao primeiro Random Forest (RF_1), cujo rótulo é considerado como resultado final da classificação.

Algorithm 2 Fluxo do Co-Training com ensino cruzado entre dois Random Forests

Require: conjunto rotulado L ; conjunto não rotulado U ; subconjuntos de atributos V_1 e V_2

Ensure: predições finais do conjunto de teste

- 1: Dividir os atributos em dois subconjuntos V_1 e V_2 .
- 2: Inicializar $L_1 \leftarrow L$ e $L_2 \leftarrow L$.
- 3: Treinar RF_1 sobre L_1 (subconjunto V_1) e RF_2 sobre L_2 (subconjunto V_2).
- 4: **repeat**
- 5: **if** $U = \emptyset$ **then**
- 6: **encerrar**
- 7: **end if**
- 8: RF_1 prevê U pelo subconjunto V_1 ; RF_2 prevê U pelo subconjunto V_2 .
- 9: Selecionar de cada RF até n instâncias com confiança $\geq$ limiar.
- 10: **if** nenhuma instância confiável foi selecionada **then**
- 11: **encerrar**
- 12: **end if**
- 13: Adicionar as confiáveis de RF_1 ao conjunto L_2 e as de RF_2 ao conjunto L_1 .
- 14: Re-treinar RF_1 sobre L_1 e RF_2 sobre L_2 .
- 15: Remover de U as instâncias já incorporadas.
- 16: **until** máximo de iterações atingido
- 17: RF_1 e RF_2 treinados e prontos para predição dos testes
- 18: **for** cada instância do conjunto de teste **do**
- 19: RF_1 e RF_2 realizam a predição utilizando, respectivamente, os subconjuntos V_1 e V_2 .
- 20: Obter a confiança associada à classe prevista por cada classificador.
- 21: **if** confiança(RF_1) $\geq$ confiança(RF_2) **then**
- 22: Predição final $\leftarrow$ predição de RF_1 .
- 23: **else**
- 24: Predição final $\leftarrow$ predição de RF_2 .
- 25: **end if**
- 26: **end for**
- 27: **return** predições finais do conjunto de teste.

Fonte: Adaptado de Williams e Qian (WILLIAMS; QIAN, 2025).

Definidos os algoritmos, esta seção descreve as condições sob as quais ambos foram treinados. Para garantir uma base de comparação estatisticamente válida, adotaram-se estratégias de amostragem uniformes para os dois métodos. Todo o processo foi conduzido de forma estratificada, utilizando uma semente aleatória (*seed*) fixa igual a 42 tanto no particionamento Holdout quanto na amostragem estratificada das frações, preservando em cada subconjunto. Esse cuidado é especialmente relevante diante do forte desbalanceamento das bases, pois evita que partições ou frações reduzidas fiquem sub-representadas em

amostras de ataque, o que comprometeria tanto o treinamento quanto a confiabilidade da avaliação.

Para avaliar a resiliência à escassez de dados, tanto o LPA quanto o *Random Forest* com *Co-Training* foram submetidos a duas condições de particionamento:

1. **Cenário padrão:** Particionamento *Holdout* estratificado, realizado apenas uma vez, com 80% das amostras para treinamento e 20% para teste [Sarantos, Violos e Leivadeas \(2025\)](#);
2. **Amostragem progressiva:** Restrição do volume de dados rotulados, no treinamento a frações de 1%, 5%, 10% e 50%, igualmente selecionadas de forma estratificada, ou seja, mantendo aproximadamente a mesma proporção de exemplos pertencentes a cada classe existente no conjunto de dados original, sendo selecionados de forma aleatória, assim permitindo analisar a eficácia do algoritmo semi-supervisionado. [Williams e Qian \(2025\)](#).

3.5.1 TRATAMENTO DO DESBALANCEAMENTO COM SMOTE

Devido ao fracionamento das amostras rotuladas, a quantidade de registros de ataque torna-se ainda mais reduzida e pouco representativa nas frações menores, prejudicando a generalização do modelo. Para mitigar esse problema, aplicou-se a técnica SMOTE (*Synthetic Minority Over-sampling Technique*) ([CHAWLA et al., 2002](#)) sobre as amostras de ataque da base de terça-feira, que gera novas instâncias sintéticas da classe minoritária por interpolação entre amostras reais e seus vizinhos mais próximos, aumentando a representatividade dos ataques sem simplesmente replicar registros existentes.

A aplicação ocorreu exclusivamente sobre o conjunto de treinamento, após o particionamento *Holdout*. Essa estratégia evita o vazamento de informações e garante que o conjunto de teste mantenha a distribuição original dos dados, permitindo uma avaliação mais confiável do desempenho.

3.6 Avaliação e Validação dos Resultados

O desempenho dos modelos foi avaliado por métricas extraídas da matriz de confusão: Acurácia, Precisão, Revocação (*Recall*) e F1-Score. Todos os modelos foram avaliados sob condições idênticas, descritas a seguir.

A separação dos dados seguiu a estratégia de *Holdout*, dividindo cada base (terça e sexta) em um conjunto de teste fixo, reservado à avaliação final, e um conjunto de treino, sobre o qual foram derivadas as frações de rotulagem. Essa partição foi gerada uma única vez e reutilizada por todos os algoritmos, de modo que o LPA e o *Random Forest* com

Co-Training enxergassem exatamente as mesmas instâncias. Assim, eventuais diferenças de desempenho decorrem do algoritmo, e não da partição dos dados.

Para reproduzir as condições do aprendizado semi-supervisionado, o conjunto de treino passou por rotulagem progressiva: em cada cenário, manteve-se rotulada apenas uma fração das amostras (1%, 5%, 10% e 50%), enquanto as demais tiveram o rótulo removido (marcado com -1). O modelo dispõe, então, de um pequeno subconjunto rotulado como referência e de um grande volume de instâncias não rotuladas, cuja estrutura é explorada para inferir os rótulos ausentes. Essa ocultação incide apenas sobre o treino, o conjunto de teste permanece integralmente rotulado em todos os cenários.

Como ambas as bases apresentam forte desbalanceamento, com grande superioridade de tráfego benigno sobre os ataques, a Acurácia isolada não é um indicador confiável, pois, um classificador que rotulasse tudo como benigno ainda obteria valores altos sem detectar uma única intrusão. Por isso, a classe de ataque foi definida como positiva em todos os relatórios. A Revocação mede quantos ataques o modelo de fato detecta, evitando falsos negativos, intrusões que passam despercebidas. A Precisão indica quantos dos alertas de ataque estão corretos, reduzindo falsos positivos que sobrecarregam os administradores. O F1-Score combina as duas em um único valor, sendo a métrica mais adequada para cenários desbalanceados como este. A avaliação foi conduzida em Python, com o *framework* Scikit-learn.

4 RESULTADOS DOS EXPERIMENTOS

Este capítulo apresenta e discute os resultados obtidos a partir da aplicação do método descrito no Capítulo 3. A Seção 4.1 caracteriza a base de dados CIC-IDS2017 em sua forma original, antes de qualquer tratamento. A Seção 4.2 apresenta a base resultante após o pré-processamento, bem como a divisão em treino e teste e as frações de rotulagem empregadas. A Seção 4.3 reúne os resultados do algoritmo *Label Propagation* (LPA) por dia e fração rotulada, a Seção 4.4 reúne os resultados do *Random Forest* com *Co-Training* sob as mesmas condições, e a Seção 4.5 traz a análise comparativa entre os dois algoritmos, discutindo desempenho preditivo, custo computacional e resiliência à escassez de rótulos.

4.1 Base de Dados

Os experimentos foram conduzidos sobre os arquivos `Tuesday.csv` e `Friday.csv`, correspondentes, respectivamente, aos dias de terça-feira e sexta-feira da base CIC-IDS2017. Esses dois dias foram selecionados por, representarem uma maior abrangência dos tipos de ataque presentes na base, além de, cenários distintos quanto à distribuição dos ataques e do tráfego. A terça-feira apresenta um cenário fortemente desbalanceado, caracterizado pelo predomínio de tráfego benigno e por uma pequena fração de ataques de força bruta. A sexta-feira, por sua vez, apresenta uma distribuição consideravelmente mais balanceada, concentrando ataques de *Botnet* no período da manhã e ataques de *Portscan*, DDoS e escaneamento de portas no período da tarde. Dessa forma, foram considerados todos os períodos disponíveis nos arquivos selecionados para a realização dos experimentos.

A Tabela 5 resume a composição original de cada base, indicando o número total de instâncias, a quantidade de amostras de ataque e benignas e o número de atributos antes do tratamento.

Tabela 5 – Resumo da base CIC-IDS2017 (antes do pré-processamento)

Dia da semana	Total	Ataques	Normais	Nº de Atributos
Terça-feira	445.909	13.835	432.074	74
Sexta-feira	703.245	288.923	414.322	74

Fonte: Elaborado pelo autor (2026).

A diferença de proporção entre as classes é evidente: na terça-feira os ataques representam cerca de 3,1% do total, enquanto na sexta-feira esse valor sobe para aproximadamente 41,1%. Esse contraste é justamente o que permite avaliar o comportamento dos algoritmos em dois regimes de desbalanceamento distintos.

4.2 Resultado do Pré-Processamento

Conforme descrito na Seção 3.3, cada base passou de forma independente por limpeza de registros com valores ausentes ou infinitos, binarização da variável alvo *Label* (0 para tráfego benigno e 1 para ataque), normalização dos atributos numéricos e ajuste do conjunto de características. A Tabela 6 apresenta a composição das bases após essas operações.

Dia

Tabela 6 – Sumário da base CIC-IDS2017 após o pré-processamento

Dia da semana	Total	Ataques	Normais	Nº de Atributos
Terça-feira	445.645	13.832	431.813	65
Sexta-feira	702.718	288.785	413.933	65

Fonte: Elaborado pelo autor (2026).

A remoção de instâncias com valores nulos ou infinitos eliminou 264 registros na base de terça-feira e 527 na de sexta-feira. O número de atributos foi reduzido de 74 para 65, em razão da exclusão de atributos de variância nula, conforme detalhado na Seção 3.3.

Em seguida, cada base foi submetida ao particionamento *Holdout* estratificado, com 80% das instâncias destinadas ao treino e 20% ao teste, preservando a proporção original entre as classes em ambos os subconjuntos. Essa partição foi gerada uma única vez e reutilizada por todos os experimentos, de modo que tanto o LPA quanto o *Random Forest* com *Co-Training* enxergassem exatamente as mesmas instâncias. A Tabela 7 apresenta o número de instâncias resultante para treino e teste.

Tabela 7 – Número de instâncias de treino e teste após o particionamento *Holdout* (80/20)

Dia da semana	Treino	Teste
Terça-feira	356.516	89.129
Sexta-feira	562.174	140.544

Fonte: Elaborado pelo autor (2026).

Sobre o conjunto de treino aplicou-se a rotulagem progressiva, mantendo-se rotulada apenas uma fração das amostras de 1%, 5%, 10% e 50%, enquanto as demais tiveram o rótulo removido (marcado com -1). A Tabela 8 mostra, para cada dia e fração, a quantidade de instâncias rotuladas e não rotuladas efetivamente utilizadas nos experimentos, conforme registrado nos arquivos de resultados.

Como descrito na Seção 3.5.1, a redução das amostras rotuladas torna a classe de ataque ainda mais escassa nas frações menores. Por isso, aplicou-se a técnica SMOTE, como evidenciado na Tabela 9, exclusivamente sobre o conjunto de treino da base de terça-feira,

Tabela 8 – Distribuição de instâncias rotuladas e não rotuladas por fração

Dia	Fração rotulada	Rotulados	Não rotulados
Terça-feira	1%	6.908	352.952
	5%	34.544	338.691
	10%	69.090	320.865
	50%	345.450	178.258
Sexta-feira	1%	5.621	556.553
	5%	28.108	534.066
	10%	56.216	505.958
	50%	281.087	281.087

Fonte: Elaborado pelo autor (2026).

após o particionamento, evitando vazamento de informação e mantendo o conjunto de teste com a distribuição original. O conjunto de teste permaneceu integralmente rotulado em todos os cenários, servindo de referência fixa para a avaliação.

Tabela 9 – Quantidade de amostras antes e depois da aplicação do SMOTE - Base Terça-Feira.

Fração	Antes do SMOTE			Depois do SMOTE		
	Ataque	Benigno	Total	Ataque	Benigno	Total
1%	111	3.454	3.565	3.454	3.454	6.908
5%	553	17.272	17.825	17.272	17.272	34.544
10%	1.107	34.544	35.651	34.544	34.544	69.088
50%	5.533	172.725	178.258	172.725	172.725	345.450

Fonte: Elaborado pelo autor (2026).

4.3 Resultados do *Label Propagation* (LPA)

4.3.1 RESULTADOS DA BASE DE TERÇA-FEIRA

A base de terça-feira é fortemente desbalanceada, com os ataques de força bruta representando apenas cerca de 3,1% das instâncias. Nesse cenário, o LPA mostrou-se altamente sensível ao volume de rótulos disponíveis. A Tabela 10 reúne as métricas obtidas para cada fração de rotulagem, tendo a classe de ataque como positiva.

A acurácia permanece alta em todas as frações (de 0,9791 a 0,9890), mas essa situação é enganosa em um cenário desbalanceado, pois, classificar quase tudo como benigno já garante valores altos, devido a baixa quantidade de ataques rotulados. Dito isso, visando o enfoque na classe positiva, a revocação e o F1-Score nos ajudam a entender melhor a situação real. Com 1% de rótulos, o LPA detecta apenas 39,9% dos ataques e atinge F1 de 0,5427. Conforme a fração aumenta, a revocação sobe até 0,7216 em 50%, e

Tabela 10 – Resultados do LPA na base de terça-feira por fração rotulada

Fração	Acurácia	Precisão	Revocação	F1-Score	Tempo (s)
1%	0,9791	0,8461	0,3995	0,5427	544,4
5%	0,9844	0,8883	0,5694	0,6940	579,6
10%	0,9863	0,8861	0,6414	0,7441	660,0
50%	0,9890	0,9044	0,7216	0,8027	1155,9

Fonte: Elaborado pelo autor (2026).

o F1 chega a 0,8027. A precisão fica sempre acima de 0,81. Por fim, o tempo de treino também cresce, de 544,4 s para 1155,9 s entre 1% e 50%.

4.3.2 RESULTADOS DA BASE DE SEXTA-FEIRA

A base de sexta-feira é consideravelmente mais balanceada (cerca de 41% de ataques), reunindo tráfego de Botnet, DDoS e escaneamento de portas. A Tabela 11 apresenta os resultados do LPA nesse cenário.

Tabela 11 – Resultados do LPA na base de sexta-feira por fração rotulada

Fração	Acurácia	Precisão	Revocação	F1-Score	Tempo (s)
1%	0,7406	0,9792	0,3769	0,5443	1320,9
5%	0,8242	0,9896	0,5784	0,7301	1349,3
10%	0,9042	0,9924	0,7727	0,8689	1342,7
50%	0,9966	0,9958	0,9959	0,9959	1335,5

Fonte: Elaborado pelo autor (2026).

Como a base é mais balanceada, a acurácia já reflete melhor o desempenho real e mostra a dependência do LPA em relação aos rótulos: vai de 0,7406 em 1% a 0,9966 em 50%. A revocação evolui de 0,3769 para 0,9959 entre as frações extremas. Com mais rótulos, o F1 sobe de 0,5443 para 0,9959. O LPA chega a um ótimo resultado em 50%, mas só com bastante supervisão. Por fim, vale ressaltar, que o tempo de treino cresce de 1.320,9 s a 1.335,5 s entre 1% e 50%.

4.4 Resultados do *Random Forest* com *Co-Training*

4.4.1 RESULTADOS DA BASE DE TERÇA-FEIRA

A Tabela 12 reúne os resultados do Co-Training na base de terça-feira, a mais desbalanceada das duas.

O contraste com o LPA é direto. Já com 1% de rótulos, o Co-Training atinge F1 de 0,9889, precisão de 0,9913 e revocação de 0,9866. Aumentar a fração traz ganhos pequenos: o F1 sobe de 0,9889 para 0,9960 entre 1% e 50%. O tempo de treino fica entre 10,8 s e 136,1 s, contra mais de mil segundos do LPA na mesma base.

Tabela 12 – Resultados do Co-Training na base de terça-feira por fração rotulada

Fração	Acurácia	Precisão	Revocação	F1-Score	Tempo (s)
1%	0,9993	0,9913	0,9866	0,9889	10,8
5%	0,9994	0,9906	0,9899	0,9902	13,7
10%	0,9996	0,9946	0,9931	0,9938	24,3
50%	0,9998	0,9949	0,9971	0,9960	136,1

Fonte: Elaborado pelo autor (2026).

4.4.2 RESULTADOS DA BASE DE SEXTA-FEIRA

A Tabela 13 apresenta os resultados do Co-Training na base de sexta-feira.

Tabela 13 – Resultados do Co-Training na base de sexta-feira por fração rotulada

Fração	Acurácia	Precisão	Revocação	F1-Score	Tempo (s)
1%	0,9966	0,9984	0,9933	0,9959	17,8
5%	0,9976	0,9992	0,9950	0,9971	20,5
10%	0,9981	0,9995	0,9960	0,9977	26,4
50%	0,9988	0,9998	0,9973	0,9986	118,9

Fonte: Elaborado pelo autor (2026).

Na base mais balanceada o resultado se repete. Com 1% de rótulos o Co-Training já alcança F1 de 0,9959, valor que o LPA só obtém com 50%. As métricas variam pouco com o aumento da fração: o F1 vai de 0,9959 a 0,9986 e a revocação de 0,9933 a 0,9973. O tempo de treino segue baixo, de 17,8 s a 118,9 s.

4.5 Análise Comparativa entre os Algoritmos

Confrontados sob condições idênticas, os dois algoritmos exibem perfis nitidamente distintos. A Tabela 14 compara diretamente as métricas F1-Score, Recall e Tempo entre as frações estipuladas em cada dia.

Tabela 14 – Comparação entre LPA e Co-Training: F1-Score, revocação de ataque e tempo de treino

Dia	Fração	F1-Score		Revocação		Tempo (s)	
		LPA	Co-Tr.	LPA	Co-Tr.	LPA	Co-Tr.
Terça-feira	1%	0,5427	0,9889	0,3995	0,9866	544,4	10,8
	5%	0,6940	0,9902	0,5694	0,9899	579,6	13,7
	10%	0,7441	0,9938	0,6414	0,9931	660,0	24,3
	50%	0,8027	0,9960	0,7216	0,9971	1155,9	136,1
Sexta-feira	1%	0,5443	0,9959	0,3769	0,9933	1320,9	17,8
	5%	0,7301	0,9971	0,5784	0,9950	1349,3	20,5
	10%	0,8689	0,9977	0,7727	0,9960	1342,7	26,4
	50%	0,9959	0,9986	0,9959	0,9973	1335,5	118,9

Fonte: Elaborado pelo autor (2026).

Os resultados da Tabela 14 demonstram que o *Co-Training* apresentou desempenho superior ao LPA em todas as frações de dados rotulados e nos dois cenários avaliados. Em relação ao F1-Score da classe de ataque, o *Co-Training* manteve valores elevados mesmo com apenas 1% de dados rotulados, enquanto o LPA apresentou desempenho inferior nas menores frações, aproximando-se do *Co-Training* apenas com maior quantidade de dados supervisionados.

Esse comportamento também é observado na revocação, em que o Co-Training manteve valores próximos de 100% em todos os experimentos, indicando maior capacidade de identificar ataques. Além disso, o tempo de treinamento foi significativamente menor para o *Co-Training* em todas as configurações, evidenciando que o método apresentou melhor desempenho preditivo aliado a uma maior eficiência computacional.

4.6 Supervisionado: Random Forest com 1% Rotulado

A fim de mencionar o ganho efetivo pela exploração dos dados não rotulados, conduziu-se um experimento de baseline no qual um classificador Random Forest foi treinado de forma *estritamente supervisionada*. Preservaram-se as mesmas condições dos demais experimentos: idêntico pré-processamento, particionamento Holdout estratificado (80/20), aplicação de SMOTE sobre o treino da base de terça-feira e a classe de ataque definida como positiva. Por fim, é possível identificar uma consistência do Random Forest mesmo com apenas 1% de dados rotulados, indicando que grande parte dos acertos está relacionada à própria capacidade do algoritmo Random Forest, e não exclusivamente ao método de treinamento com Co-Training.

A Tabela ?? apresenta os resultados obtidos com a fração mínima de 1% de rótulos, cenário mais crítico e representativo da escassez de dados rotulados encontrada em ambientes reais.

Tabela 15 – Random Forest supervisionado com 1% de dados rotulados (classe de ataque como positiva).

Dia	Acurácia	Precisão	Revocação	F1-Score	TP	FP	FN	TN
Terça-feira	0,9899	0,7731	0,9544	0,8542	2.640	775	126	85.588
Sexta-feira	0,9897	0,9809	0,9943	0,9875	57.428	1.119	329	81.668

Fonte: Elaborado pelo autor (2026).

4.7 Considerações Finais do Capítulo

Os experimentos nas bases de terça e sexta-feira da CIC-IDS2017 mostraram dois comportamentos bem diferentes. O LPA chega a bons resultados, mas depende de muitos rótulos e é sensível ao desbalanceamento, indo bem apenas com frações altas e na base

mais balanceada, além de custar caro em tempo de treino. O *Co-Training*, por outro lado, manteve F1-score e revocação de ataque superiores a 0,98 já com apenas 1% de rótulos em ambas as bases, apresentando poucos falsos negativos e baixo tempo de treinamento. Ainda assim, vale ressaltar que o modelo *baseline* também apresentou desempenho elevado com apenas 1% de amostras rotuladas. Esse resultado evidencia a capacidade do *Random Forest* de obter bom desempenho mesmo sob baixa disponibilidade de rótulos. Por fim, para o cenário de detecção de intrusão com poucos dados rotulados, o *Random Forest com Co-Training* apresentou uma relação favorável entre desempenho, robustez e custo computacional, resultado que fundamenta as conclusões apresentadas no Capítulo 5.

5 CONCLUSÃO

Neste trabalho, foi abordado o desempenho de dois algoritmos semi-supervisionados na tarefa de detecção de intrusão em redes de computadores, o *Label Propagation* (LPA) e o *Random Forest* com *Co-Training*. Ambos foram avaliados por meio da base CIC-IDS2017, o que possibilitou mensurar métricas como precisão, revocação e F1-Score para a análise de desempenho dos modelos. Para tornar o cenário o mais realístico possível, o particionamento dos dados em treino e teste foi conduzido de forma estratificada, preservando a proporção original entre as classes benigna e de ataque, e o volume de rótulos foi restringido progressivamente a frações de 1%, 5%, 10% e 50%, simulando a escassez de dados rotulados comum em ambientes reais de rede.

A partir dos experimentos realizados, as principais constatações obtidas foram:

- Um aspecto notável foi a variação de desempenho em função da fração de rotulagem. O LPA demonstrou dificuldade em classificar as amostras nas situações de treinamento com poucos rótulos, apresentando melhora à medida que a quantidade de dados rotulados aumentava. Em contrapartida, o *Random Forest* com *Co-Training* obteve excelentes resultados nas mesmas condições, não demonstrando a dependência de rótulos.
- No cenário analisado, algoritmo *Random Forest* com *Co-Training* obteve melhores resultados, superando o LPA em todas as frações de rotulagem e em ambas as bases avaliadas.
- O custo computacional observado durante os testes foi um fator relevante para a análise. O LPA, cuja complexidade é da ordem de $O(n^2)$, demonstrou dificuldade em lidar com grandes volumes de dados independentemente da fração de rótulos utilizada.

e modo geral, os resultados indicam que, para a detecção de intrusões em cenários com poucos dados rotulados, o modelo supervisionado *Random Forest* apresentou, de maneira geral, uma melhor relação entre desempenho preditivo, robustez ao desbalanceamento e eficiência computacional. A incorporação do método de treinamento *Co-Training*, por sua vez, introduzindo uma abordagem semi-supervisionada ao processo de aprendizagem permitiu observar, de forma geral, um pequeno incremento no desempenho preditivo em comparação ao modelo puramente supervisionado.

O *Label Propagation* (LPA), embora tenha alcançado resultados satisfatórios em cenários com maior proporção de dados rotulados, mostrou-se mais sensível aos efeitos do

desbalanceamento da base e apresentou maior custo computacional, fatores que podem restringir sua aplicabilidade em ambientes com grande volume de dados e disponibilidade limitada de rótulos. Dessa forma, os resultados obtidos reforçam o potencial das técnicas de aprendizado semi-supervisionado como alternativas para o aproveitamento de dados não rotulados em tarefas de detecção de intrusões, contribuindo para o desenvolvimento de abordagens voltadas à segurança cibernética e à proteção de redes de computadores.

5.1 Trabalhos Futuros

A partir dos experimentos realizados e dos resultados obtidos nesta monografia, foram identificadas diversas oportunidades para pesquisas futuras:

- Investigar estratégias que melhorem o desempenho do LPA, sobretudo em cenários fortemente desbalanceados, avaliando ajustes no número de vizinhos, no kernel de similaridade e no tratamento da classe minoritária.
- Analisar os algoritmos nos demais dias da base CIC-IDS2017, que contemplam outros tipos de ataque, a fim de verificar a robustez e a consistência dos modelos frente a diferentes perfis de tráfego;
- Buscar maneiras mais efetivas de enriquecer a base de treinamento diante da baixa quantidade de dados rotulados, explorando técnicas para aumentar a representatividade dos dados.
- Aprofundar no estudo de algoritmos semi-supervisionados além do LPA e do *Co-Training*, a fim de comparar seu desempenho e robustez na detecção de intrusão sob escassez de rótulos.
- Realizar mais testes relacionados a variação do Holdout juntamente com a diferenciação de métodos Supervisionados, quando comparados com suas variações.

REFERÊNCIAS

- ALCÂNTARA, A. O. *Comparative Study of Different Feature Sets for Intrusion Detection in Computer Networks*. 2024. Trabalho Acadêmico — UFU. Disponível em: <<https://sol.sbc.org.br/index.php/eniac/article/view/33797>>. Citado na página 6.
- ALPAYDIN, E. *Introduction to Machine Learning*. 4th. ed. Cambridge, Massachusetts: MIT Press, 2020. Citado 2 vezes nas páginas 9 e 11.
- BLUM, A.; MITCHELL, T. Combining labeled and unlabeled data with co-training. In: *Proceedings of the 11th Annual Conference on Computational Learning Theory (COLT)*. [S.l.: s.n.], 1998. p. 92–100. Citado na página 15.
- CHAPELLE, O.; SCHÖLKOPF, B.; ZIEN, A. *Semi-Supervised Learning*. Cambridge, MA: MIT Press, 2006. Citado 4 vezes nas páginas 9, 10, 12 e 14.
- CHAWLA, N. V. et al. SMOTE: Synthetic minority over-sampling technique. *Journal of Artificial Intelligence Research*, v. 16, p. 321–357, 2002. Citado na página 27.
- DIANA, L.; DINI, P.; PAOLINI, D. Overview on intrusion detection systems for computers networking security. *Computers*, MDPI, v. 14, n. 3, p. 87, 2025. Citado na página 5.
- ENGELEN, J. E. van; HOOS, H. H. A survey on semi-supervised learning. *Machine Learning*, Springer, v. 109, n. 2, p. 373–440, 2020. Citado na página 10.
- ERICSSON. *Ericsson Mobility Report*. 2023. Acesso em: 2025. Disponível em: <<https://www.ericsson.com/>>. Citado na página 5.
- GÉRON, A. *Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow*. 3. ed. [S.l.]: O'Reilly Media, 2022. Citado na página 10.
- IBM Security. *Cost of a Data Breach Report 2024*. [S.l.], 2024. Disponível em: <<https://www.ibm.com/reports/data-breach>>. Citado na página 8.
- LIPPMANN, R. et al. Evaluating intrusion detection systems. In: *DARPA Information Survivability Conference and Exposition*. [S.l.]: IEEE, 2000. Citado na página 9.
- MEIDAN, Y. et al. Cadesh: Collaborative anomaly detection for smart homes. *arXiv preprint*, 2023. Disponível em: <<https://arxiv.org/abs/2303.01021>>. Citado 2 vezes nas páginas 5 e 6.
- MITCHELL, T. M. *Machine Learning*. New York: McGraw-Hill, 1997. Citado na página 9.
- MVULA, P. K. et al. A survey on the applications of semi-supervised learning to cyber-security. *ACM Computing Surveys*, ACM, v. 56, n. 10, 2024. Citado 2 vezes nas páginas 6 e 11.
- PEDREGOSA, F. et al. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, v. 12, p. 2825–2830, 2011. Citado na página 24.

SARANTOS, P.; VIOLOS, J.; LEIVADEAS, A. Enabling semi-supervised learning in intrusion detection systems. *Computers & Security*, Elsevier, 2025. Citado 6 vezes nas páginas 6, 16, 17, 23, 24 e 27.

SCARFONE, K.; MELL, P. Guide to intrusion detection and prevention systems (IDPS): Recommendations of the National Institute of Standards and Technology. *NIST Special Publication*, v. 800-94, p. 127, 2007. Disponível em: <<http://csrc.nist.gov/publications/nistpubs/800-94/SP800-94.pdf>>. Citado na página 9.

SECURITY, I. *Cost of a Data Breach Report*. 2023. Acesso em: 2025. Disponível em: <<https://www.ibm.com/security>>. Citado na página 5.

SOMMER, R.; PAXSON, V. Outside the closed world: On using machine learning for network intrusion detection. In: *Proceedings of the 2010 IEEE Symposium on Security and Privacy (SP)*. Oakland, CA, USA: IEEE, 2010. p. 305–316. Citado 3 vezes nas páginas 9, 10 e 14.

STALLINGS, W. *Cryptography and Network Security: Principles and Practice*. 7th. ed. London: Pearson, 2017. Citado na página 8.

SYSTEMS, C. *Cisco Annual Internet Report*. 2023. Acesso em: 2025. Disponível em: <<https://www.cisco.com/>>. Citado na página 5.

VIEIRA, Y. M. *Deteção de Intrusão em Redes de Computadores Usando Técnicas de Detecção de Outliers*. 2024. Trabalho Acadêmico — UFU. Disponível em: <<https://repositorio.ufu.br/handle/123456789/41849>>. Citado na página 6.

SVIHROVÁ, R.; LETTNER, C. A semi-supervised approach for network intrusion detection. In: *Proceedings of the 15th International Conference on Availability, Reliability and Security*. New York, NY, USA: Association for Computing Machinery, 2020. (ARES '20). ISBN 978-1-4503-8833-7. Disponível em: <<https://doi.org/10.1145/3407023.3407073>>. Citado 3 vezes nas páginas 6, 15 e 17.

WILLIAMS, P.; QIAN, Y. Semi-supervised learning for intrusion detection in large computer networks. *Applied Sciences*, MDPI, v. 15, n. 5930, 2025. Citado 9 vezes nas páginas 6, 13, 15, 16, 17, 23, 24, 26 e 27.

WITTEN, I. H. et al. *Data Mining: Practical Machine Learning Tools and Techniques*. 4. ed. [S.l.]: Morgan Kaufmann, 2016. Citado na página 13.

XIANG, C.; YONG, L.; MENG, M. A novel semi-supervised learning approach for network intrusion detection. *IJCSNS International Journal of Computer Science and Network Security*, 2010. Citado na página 12.

YUAN, Y. et al. Semi-supervised tri-Adaboost algorithm for network intrusion detection. *International Journal of Distributed Sensor Networks*, SAGE Publications, v. 15, n. 6, 2019. Citado 3 vezes nas páginas 16, 17 e 22.

ZHENG, X.; YANG, S.; WANG, X. SF-IDS: An imbalanced semi-supervised learning framework for fine-grained intrusion detection. *arXiv preprint*, 2023. Citado na página 5.

ZHENG, X.; YANG, S.; WANG, X. SF-IDS: An imbalanced semi-supervised learning framework for fine-grained intrusion detection. *arXiv preprint*, 2023. Citado 3 vezes nas páginas 9, 11 e 24.

ZHU, X. *Introduction to Semi-Supervised Learning*. [S.l.]: Morgan & Claypool Publishers, 2009. (Synthesis Lectures on Artificial Intelligence and Machine Learning). Citado 2 vezes nas páginas 12 e 15.