ROGER ROBSON DOS SANTOS

APRENDIZAGEM DE MÁQUINA POR REFORÇO E FEDERADA PARA ATUALIZAÇÕES DE MODELO CONFIÁVEIS NA DETECÇÃO DE INTRUSÕES BASEADA EM REDE

CURITIBA
2025

ROGER ROBSON DOS SANTOS

APRENDIZAGEM DE MÁQUINA POR REFORÇO E FEDERADA PARA ATUALIZAÇÕES DE MODELO CONFIÁVEIS NA DETECÇÃO DE INTRUSÕES BASEADA EM REDE

Tese de Doutorado apresentada ao Programa de Pós-Graduação em Informática da Pontifícia Universidade Católica do Paraná como requisito para obtenção do título de Doutor em Informática.

Área de concentração: Ciência da Computação

Orientador: Dr. Eduardo Kugler Viegas
Coorientador: Dr. Altair Olivo Santin

CURITIBA

2025

Dados da Catalogação na Publicação
Pontifícia Universidade Católica do Paraná
Sistema Integrado de Bibliotecas – SIBI/PUCPR
Biblioteca Central
Gisele Alves – CRB 9/1578

S237a
2025

Santos, Roger Robson dos
Aprendizagem de máquina por reforço e federada para atualizações de modelo confiáveis na detecção de intrusões baseada em rede / Roger Robson dos Santos / orientador : Eduardo Kugler Viegas ; coorientador : Altair Olivo Santin. – 2025.
[106] f. : il. ; 30 cm

Tese (doutorado) – Pontifícia Universidade Católica do Paraná, Curitiba, 2025
Bibliografia: f. 99-[106]

1. Sistemas de detecção de intrusão (Segurança do computador). 2. Ciberterrorismo. 3. Aprendizado por reforço. 4. Aprendizado federado (Aprendizado de máquina). I. Viegas, Eduardo Kugler. II. Santin, Altair Olivo. III. Pontifícia Universidade Católica do Paraná. Programa de Pós-Graduação em Informática. IV. Título.

CDD. 20. ed. – 004

Pontifícia Universidade Católica do Paraná
Escola Politécnica
Programa de Pós-Graduação em Informática

Curitiba, 13 de maio de 2025.

60-2025

DECLARAÇÃO

Declaro para os devidos fins, que **ROGER ROBSON DOS SANTOS** defendeu a tese de Doutorado intitulada "**Aprendizagem De Máquina Por Reforço E Federada Para Atualizações De Modelo Confiáveis Na Detecção De Intrusões Baseada Em Rede"**, na área de concentração Ciência da Computação no dia 13 de fevereiro de 2025, no qual foi aprovado.

Declaro ainda, que foram feitas todas as alterações solicitadas pela Banca Examinadora, cumprindo todas as normas de formatação definidas pelo Programa.

Por ser verdade, firmo a presente declaração.

gov.br Documento assinado digitalmente
EMERSON CABRERA PARAISO
Data: 28/05/2025 16:57:15-0300
Verifique em https://validar.iti.gov.br

_____

Prof. Dr. Emerson Cabrera Paraiso
Coordenador do Programa de Pós-Graduação em Informática

Resumo

Nos últimos anos, as técnicas de aprendizado de máquina têm sido amplamente adotadas para detecção de intrusões em redes, especialmente devido às constantes mudanças no comportamento do tráfego de rede. No entanto, a aplicação dessas técnicas em ambientes de produção enfrenta desafios, como a necessidade de atualizações frequentes e a manutenção da confiabilidade das classificações, o que pode comprometer a segurança dos sistemas. Para enfrentar esses desafios, este trabalho propõe uma abordagem que combina Aprendizado por Reforço (RL) e Aprendizado Federado (FL). A proposta visa melhorar a confiabilidade das classificações em ambientes reais, reduzindo a necessidade de atualizações frequentes do modelo. O modelo utiliza o Aprendizado por Reforço para garantir aprendizado de longo prazo, mantendo altas taxas de classificação e confiabilidade ao longo do tempo. Além disso, as atualizações do modelo são realizadas de forma eficiente por meio de uma combinação de transfer learning e um mecanismo de janela deslizante, que diminui significativamente a necessidade de recursos computacionais e intervenção humana. A abordagem de Aprendizado Federado é adotada para minimizar o custo das atualizações, aproveitando dados locais sem centralizar dados sensíveis. Essa técnica permite identificar e rejeitar classificações imprecisas de forma não supervisionada, mesmo quando os dados estão desatualizados. Experimentos realizados com grandes volumes de dados de tráfego de rede real ao longo de anos, mostram que a nova abordagem proposta oferece taxas de precisão semelhantes aos métodos tradicionais, com a vantagem de reduzir os falsos positivos e negativos e exigir menos recursos computacionais.

Palavras-Chave: Detecção de Intrusão, Ataques de rede, Aprendizado por reforço, Aprendizado Federado.

Abstract

In recent years, machine learning techniques have been widely adopted for network intrusion detection, especially due to the constant changes in network traffic behavior. However, applying these techniques in production environments faces challenges, such as the need for frequent updates and maintaining classification reliability, which may compromise system security. To address these challenges, this work proposes an approach that combines Reinforcement Learning (RL) and Federated Learning (FL). The proposed method aims to enhance classification reliability in real-world environments while reducing the need for frequent model updates. The model leverages Reinforcement Learning to ensure long-term learning, maintaining high classification rates and reliability over time. Moreover, model updates are performed efficiently through a combination of transfer learning and a sliding window mechanism, significantly reducing the need for computational resources and human intervention. The Federated Learning approach is adopted to minimize update costs by utilizing local data without centralizing sensitive information. This technique allows for the identification and rejection of inaccurate classifications in an unsupervised manner, even when data is outdated. Experiments conducted with large volumes of real network traffic data over several years demonstrate that the proposed approach achieves accuracy rates comparable to traditional methods, with the added benefits of reducing false positives and negatives and requiring fewer computational resources.

Keywords: *Intrusion Detection, Network Attacks, Reinforcement Learning, Federated Learning.*

Lista de Ilustrações

Lista de Tabelas

Lista de Abreviaturas e Siglas

NIDS	*Network Intrusion Detection System*
FL	*Federated Learning*
RL	*Reinforciment Learning*
ML	*Machine Learning*
CTI	*Cyber Threat Intelligence*
NIC	*Network Interface Card*
FedAVG	*Average Federated Learning*
RF	*Random Forest*
kNN	*k-Nearest Neighbor*
Bag	*Bagging*
GBT	*Gradient Boosting*
TP	Verdadeiro Positivo
TN	Verdadeiro Negativo
FP	Falso Positivo
FN	Falso Negativo

Lista de Equações

Lista de Algoritmos

Sumário

Capítulo 1

Introdução

O aumento constante de ciberataques ao longo dos anos destaca a necessidade urgente de soluções de segurança mais eficazes. Relatórios recentes revelam que mais de 15% dos usuários da Internet foram alvo de ciberataques [1], com um aumento alarmante de ataques baseados em rede em 2022 [2]. Esse cenário desafia os operadores de segurança, que dependem de sistemas de detecção de intrusão de rede (*Network Intrusion Detection System* - NIDS) para identificar e neutralizar essas ameaças [3].

As abordagens tradicionais de detecção de intrusões, como as técnicas baseadas em assinatura, comparam o tráfego de rede com padrões conhecidos de ataques. Embora eficazes para ameaças conhecidas, essas técnicas falham ao lidar com novos tipos de ataques, como os de dia zero [4]. Em contraste, as abordagens baseadas em comportamento constroem perfis de tráfego normal e identificam anomalias, oferecendo uma detecção mais abrangente, inclusive para novos ataques [5]. No entanto, essas soluções dependem fortemente de modelos de aprendizado de máquina (*Machine Learning* - ML), que, apesar de promissores, apresentam desafios consideráveis [6].

Com o aumento da complexidade dos ataques e a variação dos ambientes de rede, os modelos de ML rapidamente se tornam obsoletos, exigindo atualizações frequentes e dispendiosas [6]. A natureza não estacionária dos ambientes de rede e a necessidade de retreinamento constante impõem dificuldades adicionais aos administradores de sistemas, que muitas vezes enfrentam o dilema de lidar com falsos positivos enquanto aguardam por modelos atualizados [8]. Embora a literatura reconheça a necessidade dessas atualizações, os desafios práticos de coleta de dados, rotulagem de eventos e retreinamento de modelos ainda permanecem como barreiras significativas para a implementação eficaz de soluções baseadas em comportamento [9], [10].

1.1. Motivação

Nos últimos anos, a crescente sofisticação e frequência dos ciberataques têm impulsionado o desenvolvimento de novas soluções de detecção de intrusões, particularmente aquelas baseadas em técnicas de aprendizado de máquina (ML) [3]. Esses sistemas de detecção de intrusão de rede (NIDS) são projetados para identificar comportamentos anômalos ao comparar o tráfego de rede atual com modelos previamente treinados. No entanto, o sucesso dessas soluções depende fortemente da qualidade dos dados de treinamento, que devem representar adequadamente tanto o comportamento legítimo quanto as possíveis atividades maliciosas [7], [8].

A construção de um conjunto de treinamento robusto e confiável é uma tarefa complexa, especialmente considerando a necessidade de refletir as propriedades dinâmicas e não estacionárias dos ambientes de rede [5]. Além disso, a variabilidade no comportamento do tráfego de rede ao longo do tempo exige atualizações frequentes nos modelos de ML, pois um modelo desatualizado pode resultar em uma taxa elevada de falsos positivos e falsos negativos, comprometendo a eficácia do sistema [9].

A manutenção e a atualização contínua dos modelos de ML são, portanto, essenciais para garantir que os NIDS possam enfrentar novas ameaças com precisão [10]. No entanto, o processo de atualização de modelos é repleto de desafios. A coleta de novos dados, a rotulagem precisa e o retreinamento de modelos são tarefas demoradas e intensivas em recursos [13]. Além disso, devido a preocupações com a privacidade, a colaboração entre organizações para compartilhar conjuntos de dados relevantes é limitada, dificultando ainda mais o desenvolvimento de soluções eficazes [11], [12].

As abordagens tradicionais de reconhecimento de padrões muitas vezes descartam os modelos desatualizados e constroem novos a partir do zero, sem aproveitar o conhecimento acumulado previamente [14]. Esse processo aumenta ainda mais os custos computacionais e a quantidade de dados necessários para as atualizações [10]. Como resultado, as soluções de detecção de intrusões baseadas em ML frequentemente permanecem como tópicos de pesquisa, com uso limitado em ambientes de produção [15].

Para enfrentar esses desafios, é crucial desenvolver métodos de atualização de modelos que sejam mais eficientes e capazes de integrar o conhecimento prévio, minimizando a necessidade de grandes quantidades de dados e recursos computacionais

[6]. Além disso, é necessário garantir que os NIDS mantenham sua eficácia ao longo do tempo, mesmo quando os modelos estão desatualizados, evitando a degradação do desempenho até que as atualizações possam ser implementadas [16].

A literatura atual muitas vezes negligencia a longevidade e a confiabilidade dos modelos de ML em ambientes dinâmicos, focando principalmente em alcançar altas taxas de classificação sem considerar os desafios práticos de manutenção contínua [17]. Assim, a pesquisa precisa evoluir para desenvolver soluções que não apenas detectem intrusões com precisão, mas que também sejam sustentáveis e eficazes a longo prazo, adaptando-se rapidamente às novas ameaças sem sacrificar a confiabilidade do sistema [18].

1.2. Objetivos

O objetivo geral deste trabalho é desenvolver um modelo de detecção de intrusão de rede que combine abordagens de Aprendizado Federado (*Federated Learning* - FL) e Aprendizado por Reforço (*Reinforcement Learning* - RL) para identificar classificações não confiáveis, aumentar a longevidade do modelo, reduzir a variação de precisão ao longo do tempo e facilitar as atualizações do modelo com custo reduzido. Para alcançar esse objetivo geral, os seguintes objetivos específicos devem ser atingidos:

1. Desenvolver uma Tarefa de Classificação com Opção de Rejeição: Criar um sistema que permita a rejeição de amostras com baixa confiança, garantindo que apenas amostras altamente confiáveis sejam aceitas, mesmo quando o modelo está desatualizado.
2. Definir uma Abordagem de Treinamento do Modelo Baseada em FL: Propor uma metodologia de treinamento do modelo utilizando Aprendizado Federado, focando em eventos rejeitados ao longo do tempo para melhorar a eficácia e a precisão do modelo, enquanto se mantém o custo das atualizações reduzido.
3. Elaborar um Mecanismo de Atualização do Modelo ao Longo do Tempo: Implementar um processo de atualização do modelo que utilize FL e *transfer learning* para reduzir o custo computacional e melhorar a adaptação do

modelo às novas ameaças. Isso inclui o uso de uma janela de tempo deslizante para atualizar o modelo, aproveitando o conhecimento prévio do modelo desatualizado.

4. Maximizar a Correção do Modelo com RL: Implementar um sistema de treinamento baseado em Aprendizado por Reforço que busque alta precisão e correção, medindo a distância de confiança da classificação em relação ao rótulo correto do evento. Garantir que o modelo obtenha alta correção em todos os eventos classificados, não apenas em um subconjunto durante a fase de treinamento.
5. Reduzir a Variação de Precisão ao Longo do Tempo com RL: Desenvolver um modelo de RL que se torne mais preciso e confiável ao longo do tempo, minimizando a variação de precisão e mantendo a eficácia do modelo à medida que o ambiente de rede evolui.
6. Facilitar Atualizações Eficientes do Modelo com RL: Propor um mecanismo de atualização do modelo que utilize uma janela de tempo deslizante e aproveite o conhecimento prévio do modelo desatualizado por meio de aprendizado por transferência. Reduzir os recursos computacionais e a intervenção de especialistas necessários, mantendo a eficácia do sistema de detecção de intrusões.
7. Otimizar Recursos Computacionais e Tempo de Atualização: Implementar uma abordagem de RL que minimize o uso de recursos computacionais e o tempo gasto em atualizações do modelo, evitando a construção de novos modelos do zero e estendendo significativamente a longevidade do modelo.

1.3. Contribuições

Em resumo, as principais contribuições da proposta são:

- Novo Conjunto de Dados de Intrusão: um conjunto de dados público e inédito, abrangendo quatro anos de tráfego de rede real e rotulado, com mais de 8 TB e sete bilhões de fluxos. Este recurso permite a avaliação detalhada de propostas de detecção de intrusões em ML e a análise do impacto das mudanças no comportamento do tráfego ao longo do tempo na precisão da classificação e nas atualizações do modelo.

- Avaliação Crítica das Técnicas ML: os experimentos mostram que as técnicas de ML para detecção de intrusões, amplamente utilizadas, não conseguem adaptar-se adequadamente às mudanças no tráfego de rede, resultando em um aumento na taxa de erro e variação na precisão após um ano. Demonstra-se que atualizações periódicas baseadas em novas características do tráfego são essenciais para manter a confiabilidade, embora exijam alta frequência e custo computacional.
- Modelo de Aprendizado por Reforço: um novo modelo de aprendizado por reforço para detecção de intrusões, melhorando a confiabilidade da classificação ao longo do tempo através de um ajuste fino que visa valores de confiança mais elevados. Este modelo pode prolongar a vida útil em até dois anos sem atualizações e, ao utilizar o modelo desatualizado, reduz os custos computacionais, melhorando as taxas de falsos negativos e diminuindo a variação de precisão.
- Sistema de Classificação com Rejeição: Desenvolvemos uma abordagem de classificação com rejeição que melhora a acurácia do sistema ao garantir que apenas amostras altamente confiáveis sejam aceitas, mesmo quando o modelo está desatualizado.
- Atualizações Eficientes com Aprendizado Federado (FL): Introduzimos uma abordagem baseada em FL que facilita atualizações de modelos de detecção de intrusão com menor custo computacional, aproveitando eventos de rede rotulados e reduzindo a necessidade de grandes volumes de dados novos para manter a eficácia do modelo.

1.4. Hipótese

A aplicação de técnicas combinadas de Aprendizado Federado (FL) e Aprendizado por Reforço (RL) pode melhorar significativamente a eficácia e a longevidade dos modelos de detecção de intrusão de rede.

1.5. Contribuições Científicas

Tabela 1 - Conjunto Publicações realizadas e/ou submetidas como autor principal.

Nome do Trabalho	Autores	Lugar da Publicação	Tipo	Qualis
Federated learning for reliable model updates in network-based intrusion detection	Roger R. dos Santos, Eduardo K. Viegas, Altair O. Santin e Vinicius Cogo	Computers & Security	Journal	Q1
Reinforcement learning for intrusion detection: More model longness and fewer updates	Roger R. dos Santos, Eduardo K. Viegas, Altair O. Santin e Pietro Tedeschi	IEEE Transactions on Network and Service Management	Journal	Q1
A reminiscent intrusion detection model based on deep autoencoders and transfer learning	Roger R. dos Santos, Eduardo K. Viegas, Altair O. Santin	GLOBECOM 2021	Conference	A1
Improving intrusion detection confidence through a moving target defense strategy	Roger R. dos Santos, Eduardo K. Viegas, Altair O. Santin	GLOBECOM 2021	Conference	A1
A machine learning model for detection of docker-based APP overbooking on kubernetes	Felipe Rampos, Eduardo K. Viegas, Altair O. Santin, Pedro Horchulhack, Roger R. dos Santos, Allan Epindola	ICC 2021	Conference	A1
A multi-view intrusion detection model for reliable and autonomous model updates	Rivaldo L. Tomio, Eduardo K. Viegas, Altair O. Santin, Roger R. dos Santos	ICC 2021	Conference	A1

1.6. Organização do Documento

O restante da proposta está organizado da seguinte forma. A Seção 2 descreve mais detalhadamente os desafios dos NIDS baseados em ML. A Seção 3 apresenta trabalhos relacionados sobre detecção de intrusões confiável. A Seção 4 avalia como as mudanças no comportamento do tráfego de rede afetam os NIDS baseados em ML tradicionais. A Seção 5 descreve o modelos propostos. A Seção 6 avalia os resultados obtidos com as propostas.

Capítulo 2

Fundamentação Teórica

Destacam-se os aspectos desafiadores envolvidos no projeto de um NIDS confiável baseado em aprendizado de máquina, utilizando a abordagem de aprendizado federado (FL).

2.1. Sistemas de Detecção de Intrusão

Um Sistema de Detecção de Intrusão (*Intrusion Detection System* - IDS) é uma ferramenta de segurança projetada para monitorar atividades em redes de computadores ou sistemas individuais, com o objetivo de identificar comportamentos suspeitos ou maliciosos que possam indicar a presença de um ataque de rede [19]. Existem duas categorias principais de IDS: os Sistemas de Detecção de Intrusão de Rede (*Network Intrusion Detection Systems - NIDS*) e os Sistemas de Detecção de Intrusão em Host (*Host Intrusion Detection Systems - HIDS*).

O funcionamento de um IDS envolve várias etapas críticas, desde a coleta de dados e o pré-processamento até a análise e a geração de alertas. A coleta de dados pode incluir a captura de pacotes de rede, a leitura de logs de sistema e a monitorização de atividades do usuário. O pré-processamento envolve a normalização e a filtragem dos dados para eliminar ruídos e reduzir falsos positivos. A análise pode ser realizada em tempo real ou de maneira assíncrona, utilizando algoritmos que buscam padrões suspeitos ou anômalos [20]. Quando um potencial intrusão é detectada, o IDS gera um alerta que é enviado aos administradores de rede ou a sistemas de resposta automatizada, que podem tomar medidas para mitigar a ameaça. A eficácia de um IDS depende de sua capacidade

de manter atualizados seus mecanismos de detecção e de sua integração com outras ferramentas de segurança [21].

2.1.1. Sistemas de Detecção de Intrusão de Rede

Os Sistemas de Detecção de Intrusão de Rede são ferramentas de segurança projetadas para monitorar e analisar o tráfego de rede em tempo real, com o objetivo de identificar atividades suspeitas ou maliciosas que possam indicar a presença de ataques. Esses sistemas são instalados em pontos estratégicos da rede, como em segmentos críticos ou na fronteira entre a rede interna e a Internet, permitindo a vigilância contínua do fluxo de dados que entra e sai da rede [22]. Os NIDS utilizam uma variedade de técnicas para detectar intrusões, incluindo abordagens baseadas em assinatura e em comportamento.

As técnicas baseadas em assinatura comparam o tráfego de rede observado com uma base de dados de padrões de ataque conhecidos. Quando o NIDS detecta uma correspondência com um desses padrões, ele gera um alerta para notificar os administradores de rede. Essa abordagem é eficaz na identificação de ataques previamente catalogados, mas pode falhar na detecção de novos ataques ou variações de ataques conhecidos [23]. Por outro lado, as técnicas baseadas em comportamento estabelecem um perfil de comportamento normal do tráfego de rede e identificam desvios significativos desse perfil. Essa abordagem permite a detecção de atividades desconhecidas ou anômalas que podem indicar a presença de novos tipos de ataques.

O processo de funcionamento de um NIDS envolve várias etapas, começando com a captura de pacotes de dados que circulam na rede. Esses pacotes são então pré-processados para normalizar e filtrar as informações, reduzindo a incidência de falsos positivos. A análise do tráfego pode ser realizada em tempo real, utilizando algoritmos que buscam padrões suspeitos ou anômalos [24]. Quando um potencial intrusão é detectada, o NIDS gera um alerta que pode ser enviado aos administradores de rede ou a sistemas de resposta automatizada, que podem tomar medidas para mitigar a ameaça. A eficácia de um NIDS depende de sua capacidade de atualizar continuamente seus mecanismos de detecção para acompanhar as novas ameaças e de sua integração com outras ferramentas de segurança.

2.2. Aprendizado de Máquina

A Aprendizado de Máquina (*Machine Learning* - ML) é um subcampo da inteligência artificial que envolve o desenvolvimento de algoritmos e modelos capazes de aprender e fazer previsões ou tomar decisões com base em dados. Ao invés de serem explicitamente programados para realizar uma tarefa específica, os sistemas de ML são treinados com grandes volumes de dados, permitindo que identifiquem padrões e *insights* úteis [25].

Existem várias técnicas e abordagens dentro do aprendizado de máquina, incluindo aprendizado supervisionado, não supervisionado, semi-supervisionado e por reforço. No aprendizado supervisionado, o modelo é treinado com um conjunto de dados rotulados, onde cada exemplo de entrada é emparelhado com a saída desejada [26]. O objetivo é aprender uma função que mapeie entradas para saídas, permitindo ao modelo fazer previsões para novos dados não vistos. Exemplos de algoritmos de aprendizado supervisionado incluem regressão linear, máquinas de vetores de suporte (SVM) e redes neurais.

No aprendizado não supervisionado, o modelo é treinado com dados que não possuem rótulos, e o objetivo é encontrar estruturas ou padrões ocultos nos dados. Técnicas comuns incluem agrupamento (clustering) e redução de dimensionalidade. O agrupamento, por exemplo, pode ser utilizado para segmentar clientes em grupos com comportamentos semelhantes sem conhecimento prévio das categorias. Algoritmos como k-means e análise de componentes principais (PCA) são exemplos de aprendizado não supervisionado [27]. A aprendizado semi-supervisionado combina elementos das abordagens supervisionada e não supervisionada, utilizando um pequeno conjunto de dados rotulados junto com um grande conjunto de dados não rotulados para melhorar a precisão do modelo.

A aprendizado por reforço, por outro lado, envolve treinar um agente para tomar decisões sequenciais em um ambiente dinâmico, recebendo recompensas ou penalidades com base em suas ações. O objetivo é aprender uma política que maximize a recompensa acumulada ao longo do tempo. Essa abordagem é amplamente utilizada em áreas como

robótica, jogos e sistemas de recomendação dinâmicos. Algoritmos como *Q-learning* e *deep Q-networks* (DQN) são exemplos de aprendizado por reforço [28].

O sucesso do aprendizado de máquina depende de vários fatores, incluindo a qualidade e a quantidade de dados disponíveis, a escolha dos algoritmos adequados e a capacidade de ajustar os hiperparâmetros do modelo. Além disso, é essencial validar o desempenho dos modelos em dados de teste para garantir que eles generalizem bem para novos dados. Com o avanço contínuo das técnicas de ML e a crescente disponibilidade de dados, o aprendizado de máquina continua a transformar diversos setores, oferecendo soluções inovadoras e eficientes para problemas complexos [29].

2.2.1. Aprendizado de máquina para detecção de intrusões de rede

Na prática, a implementação de ML em NIDS envolve a construção de modelos que podem classificar o tráfego de rede como benigno ou malicioso. O processo começa com a coleta de um grande conjunto de dados de tráfego de rede, que inclui exemplos de comportamentos normais e atividades maliciosas. Esses dados são divididos em conjuntos de treinamento e teste. No conjunto de treinamento, algoritmos de ML, como máquinas de vetores de suporte (SVM), redes neurais, árvores de decisão e algoritmos de floresta aleatória, são utilizados para aprender as características que distinguem o tráfego normal do tráfego malicioso [30].

O aprendizado supervisionado é amplamente utilizado para essa finalidade. Neste caso, os dados de treinamento são rotulados, permitindo que o modelo aprenda a associar padrões de tráfego com os rótulos correspondentes. Após o treinamento, o modelo pode ser validado e testado em novos conjuntos de dados para avaliar sua precisão e capacidade de generalização. Algoritmos de aprendizado não supervisionado, como clustering e técnicas de detecção de anomalias, também são usados para identificar padrões desconhecidos ou anômalos sem a necessidade de rótulos prévios [31].

Uma vantagem significativa da ML em NIDS é a capacidade de atualizar e melhorar os modelos continuamente. Conforme novos dados de tráfego são coletados, os modelos podem ser retreinados para incorporar as últimas informações sobre comportamentos normais e ataques emergentes. Isso permite que os NIDS baseados em ML se adaptem a mudanças na rede e nas táticas dos atacantes, mantendo a eficácia ao longo do tempo. Além disso, técnicas como o aprendizado incremental e a *transfer*

learning podem ser utilizadas para atualizar os modelos mais rapidamente e com menor custo computacional, aproveitando o conhecimento adquirido em iterações anteriores.

Um exemplo das etapas da construção de um de um algoritmo de NIDS utilizando ML implementado por meio quatro módulos sequenciais [3].

1) **Aquisição de dados.** Consiste na coleta de eventos de rede que trafegam pelo ambiente monitorado, como a captura de pacotes de rede de uma placa de interface de rede (*Network Interface Card* - NIC) específica.

2) **Extração de características**. Envolve a análise dos dados coletados para criar um vetor de características que descreva de forma abrangente o comportamento do evento coletado. Em sistemas de Detecção de Intrusão de Rede (NIDS), os eventos de rede geralmente são representados por fluxos de rede, que resumem o histórico de comunicação entre as entidades de rede em um determinado intervalo de tempo. Um exemplo de conjunto de características que pode ser usado para construir um fluxo de rede é mostrado na Tabela 2.

3) **Classificação**. Nesta etapa, o conjunto de características extraído é classificado e rotulado como normal ou ataque através da aplicação de um modelo de aprendizado de máquina (ML).

4) **Alerta**. Finalmente, o sistema sinaliza ao operador de rede os eventos de rede que foram classificados como ataques pelo modelo de ML, permitindo uma resposta imediata às ameaças identificadas.

A maioria das abordagens propostas segue um processo de três fases: treinamento, validação e teste. Durante a fase de treinamento, o objetivo é desenvolver um modelo de aprendizado de máquina (ML) avaliando um conjunto de dados de treinamento específico. Para garantir a eficácia dessa etapa, o conjunto de treinamento deve conter uma grande quantidade de amostras de rede que representem fielmente o comportamento observado nos ambientes de produção. A fase de validação desempenha um papel crucial no ajuste fino do modelo de ML, envolvendo a seleção de características e a otimização dos hiper parâmetros do modelo. Por fim, na fase de teste, a precisão do modelo é estimada durante a implantação do sistema no ambiente de produção.

Tabela 2 - Conjunto de características extraídas no nível da rede para cada agrupamento de características em um intervalo de janela de tempo de 15 segundos

Grupos de Características	**Características Coletadas**
Origem de Endereço de IP Origem e Destino de Endereço de IP Origem e Destino de Portas de Serviço	Número de Pacotes Número de Bytes Porcentagem de Pacotes (Flag PSH) Porcentagem de Pacotes (Flags SYN e FIN) Porcentagem de Pacotes (Flag FIN) Porcentagem de Pacotes (Flag SYN) Porcentagem de Pacotes (Flag ACK) Porcentagem de Pacotes (Flag RST) Porcentagem de Pacotes (Flag de Redirecionamento ICMP) Porcentagem de Pacotes (Flag Tempo Excedido de ICMP) Porcentagem de Pacotes (Flag de Timeout ICMP) Porcentagem de Pacotes (Flag de Outros tipos de ICMP

A construção de conjuntos de dados realistas para NIDS baseados em ML é um desafio significativo devido à complexidade e à dinâmica do ambiente de rede. Os conjuntos de treinamento devem incorporar uma variedade de características essenciais para garantir a eficácia do modelo:

- **Tráfego de rede real**. Os dados devem refletir com precisão as comunicações entre clientes e serviços de rede, representando o comportamento típico observado em ambientes de produção.
- **Protocolos de rede válidos**. O tráfego deve aderir estritamente às especificações dos protocolos de rede, conforme esperado em ambientes reais.
- **Comportamento diversificado**. O conjunto de dados deve abranger uma ampla gama de comportamentos de acordo com os protocolos de rede selecionados, refletindo a variabilidade do tráfego em ambientes de produção.
- **Ataque realista.** As amostras devem incluir uma representação adequada do comportamento de ataques, variando em tipo e estratégia.

- **Previamente rotulado**. Os eventos de rede devem ser rotulados corretamente como normais ou ataques, fornecendo uma base sólida para o treinamento do modelo.
- **Disponibilidade pública.** Os conjuntos de dados devem estar disponíveis publicamente para permitir comparações entre abordagens proposta.
- **Comportamento não estacionário.** Deve-se capturar as mudanças no comportamento do tráfego de rede ao longo do tempo, refletindo a natureza dinâmica do ambiente.

Construir conjuntos de dados realistas para NIDS baseados em ML é um desafio complexo. Operadores de rede normalmente optam por construir o conjunto de treinamento em um ambiente controlado ou coletá-lo de sua infraestrutura [32]. No entanto, um ambiente de teste controlado pode não refletir com precisão a diversidade e complexidade do tráfego de rede real [33]. Por outro lado, monitorar a infraestrutura de rede proporciona um tráfego mais autêntico, mas a rotulagem dos eventos coletados nem sempre é simples e pode não ser compartilhada publicamente [34].

Rotular adequadamente o tráfego de rede coletado representa um grande desafio para a confiabilidade dos conjuntos de dados de intrusão projetados [32]. Ambientes de teste controlados permitem rotulagem automática, mas enfrentam limitações na detecção de novas variantes de ataques [35]. Em contrapartida, rotular o tráfego de rede do mundo real em ambientes de produção é mais desafiador e, muitas vezes, requer abordagens complexas, incluindo técnicas de aprendizado não supervisionado [36]. Isso pode levar a uma alta taxa de falsos positivos [37].

Além dos desafios na construção do conjunto de dados, é importante reconhecer que o comportamento do tráfego de rede é dinâmico e evolui ao longo do tempo. Mudanças podem ser impulsionadas pela descoberta de novos ataques, introdução de novos serviços ou até mesmo alterações na infraestrutura de rede [4]. Portanto, os NIDS baseados em ML requerem atualizações periódicas do modelo para se manterem eficazes, um processo que demanda recursos e expertise para coleta e rotulagem dos dados [5].

2.2.2. Classificadores

Classificadores de machine learning são ferramentas essenciais para a análise e categorização de grandes conjuntos de dados. Eles podem ser categorizados em várias classes, baseados em árvores de decisão, e redes neurais. Classificadores lineares, como o *Support Vector Machine* (SVM), são amplamente utilizados devido à sua capacidade de maximizar a margem de separação entre classes, tornando-os robustos em problemas de classificação binária. Por exemplo, o estudo [38] apresenta o SVM como uma técnica eficaz para a classificação de padrões e reconhece seu desempenho superior em diversos conjuntos de dados. Por outro lado, classificadores baseados em árvores de decisão, como Random Forests, combinam a simplicidade das árvores de decisão com técnicas de ensemble para melhorar a precisão e reduzir o overfitting. O algoritmo do Random Forest é capaz de lidar com grandes volumes de dados e variáveis, oferecendo robustez e precisão em cenários de classificação complexos [39].

Classificadores baseados em redes neurais têm ganhado destaque devido à sua capacidade de aprender representações complexas dos dados. Em especial, as redes neurais profundas, como as *Convolutional Neural Networks* (CNNs), têm se mostrado altamente eficazes em tarefas de reconhecimento de imagem e visão computacional. Seu sucesso em competições da área é atribuído à utilização de camadas convolucionais, que permitem capturar hierarquias espaciais presentes nos dados [40]. Além disso, o uso de múltiplas camadas nas CNNs possibilita a aprendizagem de abstrações de alto nível, ampliando seu potencial de aplicação em diversos domínios. Essas abordagens ilustram a evolução dos classificadores de aprendizado de máquina, que vêm se tornando cada vez mais capazes de enfrentar desafios complexos, oferecendo soluções mais precisas e eficientes para a classificação de dados [41].

2.2.2.1. Random Forest

O Random Forest é um algoritmo de aprendizado de máquina baseado em árvores de decisão. Ele combina várias árvores de decisão para formar um "conjunto" que melhora a precisão e a robustez da classificação. Cada árvore no conjunto é construída a partir de um subconjunto aleatório dos dados de treinamento, e as previsões finais são feitas através da média ou da votação majoritária das previsões individuais das árvores. O Random Forest é eficaz na redução do *overfitting* e melhora a generalização ao

incorporar a aleatoriedade na seleção das amostras e das características, o que resulta em um modelo poderoso e versátil para diversas tarefas de classificação e regressão [42].

2.2.2.2.KNN

O *K-Nearest Neighbors* (KNN) é um algoritmo de aprendizado supervisionado que classifica uma amostra com base nas classes dos k vizinhos mais próximos no espaço de características. Este método é intuitivo e não-paramétrico, tornando-o útil para uma ampla gama de aplicações, desde a classificação de texto até a detecção de anomalias. Embora o KNN possa ser sensível à escolha de k e à escala dos dados, ele oferece uma abordagem robusta para problemas onde a estrutura dos dados é complexa e não-linear [43].

2.2.2.3.Bagging

O *Bagging*, ou *Bootstrap Aggregating*, é uma técnica de ensemble introduzida por Leo Breiman que visa melhorar a estabilidade e a precisão dos modelos de aprendizado de máquina. Ele funciona criando múltiplos subconjuntos dos dados de treinamento através do método de *bootstrap*, treinando um modelo separado em cada subconjunto, e depois combinando as previsões desses modelos através da média (para regressão) ou votação (para classificação). O *Bagging* pode reduzir a variância dos modelos baseados em árvores de decisão, resultando em um desempenho significativamente melhor em termos de precisão preditiva, especialmente em conjuntos de dados ruidosos [44].

2.2.2.4.Emsemble

O ensemble de votação é uma técnica simples e eficaz para melhorar a precisão dos modelos de aprendizado de máquina combinando previsões de múltiplos classificadores. Neste método, vários modelos são treinados separadamente no mesmo conjunto de dados, e suas previsões são combinadas através de uma proposta de votação. Existem duas abordagens principais: votação majoritária (onde cada modelo tem um voto

igual e a classe com mais votos é escolhida) e votação ponderada (onde diferentes pesos são atribuídos a cada modelo com base na sua precisão [45].

Essa abordagem é particularmente útil quando os modelos individuais são complementares, ou seja, quando cada um deles captura diferentes aspectos dos dados. A simplicidade e a eficácia da votação tornam-na uma técnica popular para melhorar a robustez e a precisão dos sistemas de aprendizado de máquina, especialmente em aplicações práticas onde a combinação de múltiplos modelos pode levar a uma melhora significativa no desempenho preditivo.

2.3. Aprendizado Por Reforço (*Reinforcement Learning* - RL)

Ao contrário das técnicas tradicionais de aprendizado de máquina (ML), o aprendizado por reforço (RL) visa encontrar uma estratégia de política de aprendizado ótima durante a fase de construção do modelo [46]. As técnicas baseadas em RL são executadas por meio de uma entidade chamada agente. O agente realiza suas ações em um determinado ambiente através de mecanismos de percepção e ação. O primeiro é usado pelo agente para coletar o estado atual do ambiente, enquanto a ação permite que o agente atue no ambiente dado.

Geralmente, uma abordagem baseada em RL é executada iterativamente sobre o ambiente durante a fase de treinamento. Em cada iteração, o agente recebe como entrada o estado atual do ambiente, medido através do vetor de características. Pode-se notar que o agente atua através de uma ação sobre o ambiente, conforme definido por sua política (por exemplo, aplicando um modelo de ML sobre o estado do ambiente), o que altera seu estado e gera uma nova recompensa como saída. Portanto, o objetivo do procedimento de treinamento é encontrar um agente que realize ações que aumentem suas recompensas obtidas ao longo do tempo.

O ML tradicional depende de um par de entrada e saída, onde um modelo de ML recebe um vetor de características de um evento como entrada e fornece seu valor de classe estimado como saída. As abordagens baseadas em RL são significativamente diferentes das baseadas em ML tradicional [46]. As técnicas baseadas em RL não recebem o valor da classe do evento [10]. Em vez disso, após uma ação ser realizada, o agente recebe apenas a recompensa por sua ação e o estado subsequente do ambiente. O agente aprende o limiar de decisão ideal a longo prazo, dado que otimiza suas recompensas ao longo do tempo. Notavelmente, a detecção de intrusões através de técnicas de RL ainda

está em desenvolvimento [47], mas são muito promissoras. Autores de soluções na literatura geralmente buscam maior precisão de classificação modelando o campo de detecção de intrusões como um ambiente baseado em RL[47].

2.4. Aprendizado Federado (*Federated Learning* - FL)

O aprendizado federado representa uma abordagem inovadora na área de aprendizado de máquina distribuído, onde os modelos são treinados localmente em dispositivos distribuídos, enquanto os dados permanecem nos locais de origem, mantendo assim sua sensibilidade e privacidade [48]. Esta abordagem oferece uma solução promissora para lidar com conjuntos de dados distribuídos e sensíveis, encontrados em muitos cenários do mundo real, como dispositivos móveis e ambientes de saúde. Ao preservar a privacidade dos dados, o aprendizado federado permite a colaboração entre entidades sem a necessidade de compartilhar informações sensíveis, o que é fundamental em um mundo cada vez mais preocupado com a segurança da informação.

O aprendizado federado tem sido cada vez mais reconhecida como uma abordagem essencial para lidar com os desafios de privacidade e segurança associados ao treinamento de modelos em dados sensíveis, como informações médicas e financeiras [49]. Ao permitir que os modelos sejam treinados localmente em dispositivos distribuídos, o aprendizado federado oferece uma maneira de aproveitar os *insights* valiosos contidos nos dados sem comprometer a privacidade dos indivíduos ou a confidencialidade das informações.

Além disso, o aprendizado federado tem implicações significativas para aplicações práticas, especialmente em setores onde a proteção dos dados é uma preocupação primordial. Por exemplo, em sistemas de saúde, onde os dados do paciente são altamente sensíveis, o aprendizado federado permite o desenvolvimento de modelos de aprendizado de máquina para diagnóstico médico e previsão de doenças sem comprometer a privacidade dos pacientes [50]. Essa capacidade de colaborar e extrair *insights* de dados distribuídos sem comprometer a segurança ou a privacidade é fundamental para o avanço de tecnologias baseadas em dados em vários domínios.

Um exemplo de FL, são os treinamentos de modelos em dispositivos locais, como smartphones ou dispositivos IoT, sem a necessidade de enviar dados brutos para um servidor central [51]. Em vez disso, o modelo é enviado para os dispositivos dos usuários, onde são treinados com dados locais. Após o treinamento, apenas os parâmetros do modelo são enviados de volta ao servidor central, onde são agregados para atualizar o modelo global [52]. Esse processo permite que os modelos sejam treinados de forma colaborativa, mantendo a privacidade dos dados dos usuários.

Uma técnica de FL encontrada na proposta é o FedAVG (*Average Federated Learning*), que é uma abordagem de agregação de parâmetros usada para atualizar o modelo global. No FedAVG, os modelos locais são treinados nos dispositivos dos usuários e, em seguida, os parâmetros do modelo são enviados de volta para o servidor central. No servidor, esses parâmetros são agregados computando-se a média dos parâmetros de todos os dispositivos participantes. Essa média ponderada é então usada para atualizar o modelo global. O FedAVG é eficaz porque permite que os dispositivos locais treinem modelos personalizados de acordo com seus próprios dados, enquanto mantém a acurácia do modelo global.

Uma vantagem significativa do FL, juntamente com o FedAVG, é a capacidade de preservar a privacidade dos dados dos usuários. Como os dados permanecem nos dispositivos locais e apenas os parâmetros do modelo são compartilhados, os usuários têm mais controle sobre suas informações pessoais. Isso é especialmente importante em cenários sensíveis à privacidade, como segurança, saúde ou finanças, onde os dados do usuário são altamente confidenciais. Além disso, o FL também reduz a carga de comunicação, já que apenas os parâmetros do modelo são transmitidos pela rede, em vez de grandes conjuntos de dados.

$$w_G^{t+1} = \frac{\sum_{i=1}^{m} w_i^t}{m}$$

Equação 1

Para alcançar esse objetivo, o FL é geralmente implementado seguindo um processo de quatro fases, a saber: Inicialização, Distribuição, Treinamento Local e Agregação conforme apresentado na equação 1.

1) Inicialização. O servidor central S inicializa um modelo de classificação global selecionado $\boldsymbol{W^t_G}$, onde "$\boldsymbol{t}$" denota a rodada de execução. O modelo selecionado deve permitir que a tarefa de agregação seja conduzida, por exemplo, usando uma rede neural para agregar os pesos computados.

2) Distribuição. Em cada rodada de comunicação t, tal que $0 \leq t \leq T$, onde "T" denota o limite superior de rodadas de comunicação, o servidor central S distribui o modelo global $W_G{}^t$ para um número selecionado m de pares, de modo que $0 \leq m \leq k$, onde k denota o número total de pares participantes.

3) Treinamento Local. Cada par previamente selecionado P_i realiza o treinamento local do modelo global recebido W^t_G com base em seus dados de treinamento privados D_i, produzindo um modelo local $w_i{}^t$.

4) Agregação. O servidor central S coleta os modelos locais de cada par selecionado para realizar a tarefa de agregação do modelo que compõe um novo modelo global $W_G{}^{t+1}$. A tarefa de agregação do modelo é uma função que combina uma série de modelos em um único, por exemplo, por meio da função FedAVG [53], implementada com base na seguinte equação: Equação 1 onde m denota o número de pares selecionados. Portanto, o FedAVG constrói o novo modelo global $W_G{}^{t+1}$ computando a média dos pesos de cada modelo local $W_i{}^t$. Finalmente, se a rodada de execução $t + 1$ atingir o limite superior de rodadas "T", o treinamento é encerrado; caso contrário, uma nova fase de Distribuição ocorre.

Como resultado, pensando no cenário de segurança da informação onde os dados são sensíveis e as organizações não gostariam de compartilhar, o processo de treinamento FL pode construir um modelo de aprendizado de máquina robusto com base nos dados de treinamento de cada organização, ao mesmo tempo em que mantém esses dados privados durante esse processo. A razão para isso é que o processo de treinamento é realizado localmente com base nos dados privados de cada organização, enquanto a agregação requer apenas o compartilhamento dos modelos locais construídos.

2.5. Classificador de Rejeição de Eventos de Baixa Confiança

A técnica de rejeição, no contexto da segurança da informação, é uma abordagem fundamental para mitigar riscos e proteger sistemas contra ameaças cibernéticas. Ao aplicar essa técnica em modelos de detecção de intrusões, por exemplo, o objetivo é reduzir a taxa de falsos positivos, ou seja, evitar que atividades normais sejam identificadas como intrusões. A rejeição pode ser implementada estabelecendo-se um

limite de confiança para as decisões do modelo. Esse limite é baseado em métricas como a probabilidade de uma instância ser classificada corretamente ou a incerteza associada à sua previsão. Instâncias que não atingem esse limiar são rejeitadas, permitindo que sejam submetidas a uma análise adicional antes de serem consideradas uma ameaça.

Na prática, a técnica de rejeição é amplamente aplicada em sistemas de detecção de intrusões, onde a precisão é crucial para identificar e responder a atividades maliciosas. Por exemplo, em firewalls de próxima geração, a rejeição de tráfego de rede com base em assinaturas conhecidas e padrões de comportamento suspeitos ajuda a reduzir falsos positivos e evitar o bloqueio indevido de tráfego legítimo.

Um exemplo prático, no qual um sistema de detecção de intrusões em redes de computadores que classifica o tráfego de rede como "Normal" ou "Ataque". Após a classificação, aplicamos a técnica de rejeição com um limite de confiança de 0,7. Se o sistema atribuir uma probabilidade de ser "Normal" maior que 0,7, aceitamos a classificação. Por exemplo, se uma instância tem probabilidade 0,9 de ser "Normal", ela é aceita. No entanto, se a probabilidade for menor que 0,7, rejeitamos a classificação e investigamos mais a fundo. Por exemplo, se a probabilidade for 0,6, a instância é rejeitada para análise adicional. Isso nos ajuda a reduzir falsos positivos, protegendo melhor a rede contra atividades suspeitas.

Após aplicar a rejeição, é calculado o número de instâncias corretamente classificadas como "Normal" e o número de instâncias classificadas incorretamente como "Normal" (falsos positivos). Esses valores nos ajudarão a construir a curva de Pareto, que mostra a relação entre a taxa de verdadeiros positivos e a taxa de falsos positivos para diferentes limiares de confiança.

Para traçar a curva de Pareto, é variado o limite de confiança de 0 a 1 e repetido o processo de rejeição e cálculo das taxas de verdadeiros e falsos positivos para cada valor do limite. Isso nos permitirá visualizar como o desempenho do modelo é afetado pelo ajuste do limite de confiança.

Finalmente, é plotado os resultados em um gráfico, onde o eixo x representará a taxa de falsos positivos e o eixo y representará a taxa de verdadeiros positivos. Essa representação nos fornecerá uma visão clara de como o desempenho do modelo varia com diferentes limiares de confiança, ajudando-nos a escolher o valor ideal para o limite de rejeição.

2.6. Considerações Finais

Em um mundo cada vez mais conectado, a segurança da informação é uma preocupação fundamental para empresas e organizações que lidam com grandes volumes de dados. Nesse contexto, os sistemas de detecção de intrusão desempenham um papel vital na proteção contra ameaças cibernéticas, identificando e respondendo a atividades maliciosas em tempo real. A integração de técnicas de aprendizado de máquina nesses sistemas permite uma detecção mais precisa e adaptativa, capaz de identificar padrões complexos de comportamento malicioso.

A abordagem do aprendizado federado surge como uma solução promissora para lidar com os desafios de privacidade e escalabilidade enfrentados pelos sistemas de detecção de intrusão. Ao permitir que modelos de aprendizado de máquina sejam treinados localmente em dispositivos distribuídos, sem a necessidade de compartilhamento de dados sensíveis, o aprendizado federado mantém a privacidade dos dados enquanto aproveita os *insights* coletivos de toda a rede. Além disso, a técnica de rejeição oferece um mecanismo eficaz para reduzir falsos positivos e melhorar a precisão da detecção de intrusões, garantindo assim uma defesa robusta contra ameaças cibernéticas em ambientes distribuídos e heterogêneos.

Capítulo 3

Trabalhos Relacionados

Neste contexto de aprendizado de máquina para a detecção de intrusões em redes e aprendizado federado, é essencial explorar os trabalhos relacionados que contribuíram para o avanço dessas áreas. A pesquisa e o desenvolvimento em detecção de intrusões baseada em ML com aprendizado federado têm sido fundamentais para enfrentar os desafios da segurança cibernética em um mundo cada vez mais interconectado. Esta seção busca contextualizar o atual estado da arte, examinando estudos anteriores que abordaram questões semelhantes, soluções propostas e as contribuições relevantes de pesquisadores na área. Ao revisitar esses trabalhos, é possível identificar tendências, lacunas de pesquisa e as inovações mais recentes que moldam a evolução desses campos críticos da segurança de redes e privacidade de dados.

3.1. Confiabilidade da Detecção de Intrusão

Nas últimas décadas, vários trabalhos têm proposto técnicas de aprendizado de máquina altamente precisas para detecção de intrusões em redes [3], [60]. Apesar dos resultados promissores relatados na literatura científica, as abordagens propostas dificilmente são implantadas em ambientes de produção, permanecendo principalmente como um tópico de pesquisa [58]. Em geral, os trabalhos relacionados não abordam os desafios de suas abordagens propostas quando precisam ser implantados em produção, como a mudança do comportamento do tráfego de rede e o processo periódico de atualização do modelo [4].

Em um estudo realizado por Saba et al. [54], uma notável abordagem de aprendizado profundo para a detecção de intrusões em dispositivos com recursos limitados foi apresentada. Esta abordagem inovadora demonstrou uma melhoria significativa na precisão quando comparada a outras técnicas de detecção de intrusões, mesmo em um conjunto de dados desatualizado. No entanto, é importante ressaltar que durante a avaliação desse método, não foram levados em consideração os

comportamentos altamente variáveis e não estacionários que ocorrem comumente nas redes no cotidiano. Em contrapartida, o estudo conduzido por Zeng et al. [55] concentrou-se na resolução do desafio da confiabilidade na detecção de intrusões, visando uma maior capacidade de generalização do modelo. Os autores propuseram um sistema de aprendizado profundo causal e estável, projetado especificamente para identificar relações causais entre diversos conjuntos de dados de intrusão. O resultado desse esforço foi um modelo que alcançou taxas de precisão mais consistentes em diferentes conjuntos de dados, assegurando uma sólida generalização do modelo em ambientes variados. Contudo, é importante notar que, infelizmente, os autores desse estudo também não consideraram as mudanças na relação causal ao longo do tempo, tratando o tráfego de rede como uma entidade estática em suas avaliações. Portanto, embora ambas as abordagens tenham contribuído substancialmente para o campo da detecção de intrusões, existe uma necessidade premente de futuras investigações que levem em conta a dinâmica das redes ao longo do tempo, a fim de aprimorar ainda mais a eficácia desses modelos.

Zhao et al. [56] abordaram de maneira eficaz o desafio do comportamento não estacionário em sistemas de detecção de intrusões, empregando uma abordagem inovadora baseada na detecção fora da distribuição. Neste estudo, os autores desenvolveram um sistema que leva em consideração não apenas os comportamentos conhecidos, mas também os comportamentos desconhecidos em relação ao modelo previamente implantado. Isso permitiu a identificação de novos eventos de intrusão de forma mais robusta e eficiente. No entanto, é importante observar que, assim como os estudos anteriores, esta pesquisa não abordou as mudanças dinâmicas no comportamento do tráfego de rede ao longo do tempo e o desafio de manter o modelo atualizado.

Wahab et al. [57] abordaram o problema das mudanças no tráfego de rede adotando um mecanismo de detecção de mudança de conceito. O modelo proposto foi projetado para operar em um ambiente supervisionado, onde os rótulos dos eventos podem ser fornecidos conforme o tempo avança. No entanto, é importante notar que essa condição de ambiente supervisionado pode ser considerada irrealista em muitos cenários de produção, onde a rotulagem contínua dos eventos pode ser desafiadora ou até mesmo impossível de obter de maneira precisa. Portanto, embora essa abordagem tenha proporcionado *insights* valiosos sobre a detecção de mudanças no tráfego de rede, ainda

há espaço para investigações adicionais que abordem os desafios da rotulagem em ambientes de intrusão do mundo real.

Ahmim et al. [58] apresentam uma abordagem inovadora na detecção de intrusões, propondo um conjunto de classificadores baseados em árvores de decisão. Sua estratégia demonstrou um aprimoramento notável na precisão em comparação com o uso de classificadores individuais. No entanto, é essencial ressaltar que, embora tenham alcançado melhorias na precisão, a questão crucial da atualização contínua do modelo foi deixada de lado. Este desafio de manter o modelo eficaz e adaptável ao longo do tempo representa uma lacuna importante a ser preenchida na pesquisa.

Outra abordagem que merece destaque é a proposta por J. Kevric et al. [59], que também utiliza um conjunto de classificadores baseados em árvores. Neste caso, os autores conduzem uma avaliação abrangente de diferentes técnicas de combinação de classificadores. No entanto, assim como no estudo anterior, as atualizações do modelo não são abordadas. A ausência de consideração para a evolução do modelo ao longo do tempo é uma limitação significativa dessas abordagens.

Técnicas de classificação baseadas em redes neurais convolucionais têm se destacado na detecção de intrusões. Por exemplo, R. Vinayakumar et al. [60] demonstraram que redes neurais convolucionais proporcionam detecções mais precisas em comparação com técnicas de classificação rasas. De maneira semelhante, H. Yang et al. [61] confiam nas redes neurais convolucionais para a extração de características relacionadas a ataques em redes sem fio, obtendo melhorias na precisão da detecção em relação às abordagens rasas. No entanto, é surpreendente que, apesar do sucesso dessas técnicas baseadas em redes profundas, o desafio crítico da atualização contínua do modelo seja frequentemente negligenciado na pesquisa. H. Yang et al. [61], por exemplo, propõem uma abordagem de *transfer learning* em redes neurais para a detecção de intrusões e abordam as atualizações do modelo. No entanto, sua suposição de que novos serviços podem ser prontamente identificados e rotulados conforme necessário é problemática em sistemas de detecção de intrusões em redes.

A aplicação de *autoencoders* profundos também tem ganhado destaque em trabalhos recentes em Sistemas de Detecção de Intrusões em Redes (NIDS). X. Li et al. [62] empregam um *autoencoder* profundo juntamente com uma abordagem de seleção de características utilizando arvores aleatórias para reduzir os custos computacionais associados às atualizações do modelo. Entretanto, os autores assumem que os eventos podem ser rotulados à vontade, desconsiderando a quantidade real de instâncias rotuladas

necessárias nas atualizações do modelo. Em contraste, Y. Yan et al. [63] aplicam uma série de *autoencoders* para a tarefa de extração de características, obtendo melhorias na precisão. A pesquisa futura precisa se concentrar em abordar esse desafio fundamental para tornar essas técnicas de detecção de intrusões mais robustas e adaptáveis às mudanças constantes no ambiente de segurança cibernética.

3.2. Atualização do Modelo

Diversos estudos na literatura, incluindo referências como [62], têm como premissa a suposição de que a taxa de precisão durante a fase de teste permanece constante ao longo do tempo. Quando se trata de considerar as atualizações do modelo, muitos pesquisadores adotam abordagens de aprendizado incremental para enfrentar o desafio da evolução do comportamento do tráfego de rede.

Mahdavi et al. [64] apresentaram uma abordagem de aprendizado ativo que se apoia em atualizações incrementais do modelo, buscando lidar com as mudanças no comportamento do tráfego de rede. Eles identificaram novos padrões de comportamento com base em agrupamentos de eventos previamente conhecidos, presumindo que o novo tráfego será substancialmente distinto. No entanto, é fundamental observar que o comportamento do tráfego de rede pode não apresentar variações significativas ao longo do tempo, mesmo quando novos padrões surgem. Da mesma forma, Huang et al. [65] introduziram o aprendizado incremental na detecção de intrusões para incorporar novos comportamentos de ataques. Embora sua proposta seja eficaz em lidar com novos tipos de ataques ao longo do tempo, ela não aborda a questão de como identificar esses novos padrões de comportamento no tráfego de rede.

Jiang et al. [66] propuseram o aprendizado incremental em conjunto, visando aprimorar a precisão das previsões. No entanto, os autores presumem que é simples sinalizar novos padrões de comportamento no tráfego, sem considerar os desafios associados à identificação desses novos padrões de comportamento de maneira precisa. Wu et al. [67] seguiram uma abordagem de ajuste incremental de um conjunto de classificadores com base em dados recentes. Sua estratégia mostra potencial para melhorar a precisão na classificação de novos lotes de dados, mas não se aprofunda na

identificação precisa dos novos padrões de comportamento no tráfego de rede. Li et al. [63] lidaram com as atualizações do modelo de detecção de intrusões por meio de uma abordagem de *Transfer Learning*. Eles utilizaram o modelo desatualizado como ponto de partida para reduzir os custos computacionais durante as atualizações do modelo. No entanto, a proposta de Li et al. não abordou explicitamente a questão da identificação precisa dos novos padrões de comportamento no tráfego de rede.

3.3. Aprendizado Por Reforço para Detecção de Intrusão

Recentemente, uma abordagem promissora na detecção de intrusões tem se baseado no aprendizado por reforço (RL). Suwannalai e Polprasert [68] propuseram um proposta de RL profundo para aumentar a precisão em um único conjunto de dados. Os autores forneceram maior precisão de detecção em comparação com as técnicas tradicionais de ML, mas negligenciaram os desafios relacionados à implantação real de seu modelo proposto em ambientes de produção, como as mudanças no comportamento do tráfego de rede. Benaddi et al. [69] propuseram uma proposta de RL para detectar adversários maliciosos em redes de sensores sem fio. A abordagem proposta por eles conseguiu aumentar a precisão quando comparada às técnicas tradicionais de ML. No entanto, as atualizações do modelo também foram negligenciadas.

Outra técnica de RL foi proposta por Servin e Kudenko [70] para aprendizado distribuído em detecção de intrusões. O modelo proposto por eles aborda atualizações de modelo de maneira distribuída, seguindo uma estratégia baseada em hierarquia. Infelizmente, o alívio da carga de atualização do modelo foi negligenciado, pois eles assumem que o rótulo dos eventos pode ser solicitado conforme necessário. Lopez-Martin et al. [47] propuseram uma aplicação mais prática de RL profundo. Em seu trabalho, os autores propõem um novo procedimento de treinamento para RL que substitui o ambiente ao vivo por mini-lotes de um conjunto de dados de treinamento. Assim, é viável para atualizações de modelo que seguem tal procedimento e aproveitam o modelo desatualizado. Infelizmente, os autores negligenciam as atualizações do modelo em sua avaliação e como o procedimento de atualização do modelo pode ser facilitado diante de novos comportamentos de tráfego de rede.

3.4. Aprendizado Federado para Detecção de Intrusão

Na literatura, foram propostas diversas abordagens de Aprendizado Federado (FL) para enfrentar os desafios no contexto da detecção de intrusões em redes. Por exemplo, o estudo realizado por Yuan et al. [71] teve como objetivo não apenas reduzir as sobrecargas de comunicação ao compartilhar aprendizado federado, mas também aprimorar a eficiência do processo. Para atingir esse objetivo, os autores desenvolveram um modelo inovador baseado em impulsionamento de gradiente e árvores de decisão, o qual demonstrou a capacidade de reduzir significativamente os custos de comunicação, mantendo uma alta precisão na detecção de intrusões. No entanto, é crucial notar que a pesquisa não abordou o processo de atualização do modelo ao longo do tempo, nem como o modelo se adapta a mudanças no comportamento do tráfego de rede, deixando espaço para investigações futuras nesse aspecto crítico. Outra abordagem notável, que busca facilitar a agregação de modelos, foi apresentada por Sun et al. [72]. Antes de criar um modelo global, os autores optaram por desenvolver modelos locais em segmentos individuais de rede. Essa estratégia pode efetivamente reduzir os custos associados à agregação de modelos, mas, infelizmente, não aborda a gestão do processo de atualização do modelo ao longo do tempo e não considera a confiabilidade das classificações resultantes.

Em termos gerais, o FL tem sido amplamente adotado na detecção de intrusões para permitir que os participantes compartilhem conhecimento sobre intrusões sem comprometer a privacidade de seus dados. Mothukuri et al. [73] introduziram um modelo de detecção de intrusões baseado em FL que possibilita que dispositivos com recursos limitados compartilhem conhecimento local de dados sem revelar o conteúdo dos dados em si. Este modelo alcançou resultados de acurácia comparáveis às abordagens tradicionais globalizadas. Em contraste, He et al. [74] propuseram uma abordagem inovadora que aplica a tecnologia blockchain para armazenar os pesos locais do modelo de forma confiável ao longo do tempo. Essa abordagem oferece a capacidade de realizar filtragem de alertas de maneira eficaz. No entanto, vale mencionar que esta pesquisa também não abordou de forma abrangente o processo de atualização do modelo e sua confiabilidade em cenários de detecção de intrusões em constante evolução. Portanto, há

uma necessidade contínua de investigações adicionais para aprimorar a robustez e a eficácia desses modelos em ambientes dinâmicos de segurança de rede.

3.5. Considerações Finais

Tabela 3 - Comparação dos trabalhos relacionados.

Trabalho	**Tratam Mudança de Comportamento?**	**Tratam Atualização do Modelo?**	**Tratam Confiabilidade do Sistema?**	**Tratam o custo de Rotulagem dos dados**
Saba et al. [54]	Não foi abordado	Não foi abordado	Sim, por meio de uma abordagem com CNN.	Não foi abordado
Zeng et al. [55]	Não foi abordado	Não foi abordado	Sim, utilizando aprendizado profundo para identificar relações casuais nos conjuntos de dados.	Não foi abordado
Zhao et al. [56]	Não foi abordado	Não foi abordado	Sim, utilizando uma abordagem de detecção fora da distribuição dos dados do conjunto.	Não foi abordado
Wahab et al. [57]	Sim, por meio de uma abordagem para ambientes supervisionados.	Não foi abordado	Sim, por meio de uma abordagem que se preocupa com data drift.	Não foi abordado
Ahmim et al. [58]	Não foi abordado	Não foi abordado	Sim, por meio de um conjunto de árvores de decisão.	Não foi abordado

Kevric et al. [59]	Não foi abordado	Não foi abordado	Sim, também, por meio de um conjunto de arvores de decisão	Não foi abordado
Vinayakumar et al. [60]	Não foi abordado	Não foi abordado	Sim, Por meio de redes neurais convolucionais	Não foi abordado
Yang et al. [61]	Não foi abordado	Não foi abordado	Sim, por meio de extração de características junto as redes neurais convolucionais.	Não foi abordado
Li et al. [62]	Não foi abordado	Não foi abordado	Sim, por meio de autoenconders e uma abordagem de seleção de características utilizando arvores aleatórias.	Não foi abordado
Yan et al. [63]	Não foi abordado	Não foi abordado	Sim, por meio de autoenconders na tarefa de extração de características.	Não foi abordado
Mahdavi et al. [64]	Não foi abordado	Sim, por meio de aprendizado ativo com atualizações incrementais	Sim, sua abordagem é capaz funcionar mesmo em ambientes desatualizados sem mudança de comportamento.	Não foi abordado

Huang et al. [65]	Não foi abordado	sim, por meio de aprendizado incremental para incorporar novos comportamentos	Sim, sua abordagem é capaz funcionar mesmo em ambientes desatualizados que não ajam mudança de comportamento.	Não foi abordado
Jiang et al. [66]	Não foi abordado	Sim, por meio de aprendizado incremental em conjunto	Não foi abordado	Não foi abordado
Wu et al. [67]	Não foi abordado	Sim, por meio uma abordagem com ajuste incremental em um conjunto de classificadores	Não foi abordado	Não foi abordado
Yuan et al. [71]	Não foi abordado	Sim, utilizando aprendizado federado	Sim, por meio de uma abordagem de FL com impulsionamento de gradiente e árvores de decisão.	Não foi abordado
Sun et al. [72]	Não foi abordado	Não foi abordado	Sim, por meio de uma abordagem de FL capazes de desenvolver modelos locais.	Não foi abordado
Mothukuri et al. [73]	Sim, por meio de uma abordagem de FL com modelos locais, capazes de adaptar ao comportamento.	Não foi abordado	Não foi abordado	Não foi abordado

Hei et al. [74]	Sim, por meio de uma abordagem de FL para armazenar os pesos locais do modelo de forma confiável ao longo do tempo.	Não foi abordado	Não foi abordado	Não foi abordado
Suwannalai e Polprasert [68]	Não foi abordado	Não foi abordado	Sim, por meio de uma abordagem de RL capazes de desenvolver modelos locais.	Não foi abordado
Benaddi et al. [69]	Não foi abordado	Não foi abordado	Sim, por meio de uma abordagem de RL capazes de desenvolver modelos locais.	Não foi abordado
Servin e Kudenko [70]	Não foi abordado	Sim, por meio de uma abordagem de RL.	Não foi abordado	Não foi abordado
Lopez-Martin et al. [47]	Não foi abordado	Sim, por meio de uma abordagem de RL.	Não foi abordado	Não foi abordado

Embora tenham surgido diversas abordagens promissoras na detecção de intrusões em redes, a maioria dos estudos enfrenta desafios cruciais na transição para ambientes de produção, especialmente em relação à atualização contínua do modelo. Embora abordagens como o aprendizado profundo, a detecção fora da distribuição e o uso de autoencoders tenham proporcionado melhorias significativas na precisão da detecção. Contudo é necessário avaliar estas abordagens ao longo do tempo, sendo lacunas que precisam ser preenchidas pela pesquisa futura para garantir a eficácia e a adaptabilidade desses modelos em ambientes do mundo real.

Enquanto o tratamento das atualizações do modelo na detecção de intrusões revela uma lacuna significativa na identificação precisa dos novos padrões de comportamento no tráfego de rede ao longo do tempo. Embora diversos estudos tenham adotado abordagens de aprendizado incremental para lidar com a evolução do comportamento do tráfego, muitos deles presumem uma distinção clara entre o tráfego antigo e o novo, o que pode não refletir com precisão a realidade dos ambientes de rede em constante mudança. Essa limitação destaca especialmente em cenários onde a evolução do tráfego de rede é contínua e sutil.

Em resumo, embora abordagens de aprendizado por reforço (RL) tenham mostrado potencial na melhoria da precisão na detecção de intrusões, várias propostas, como as de Suwannalai e Polprasert [68], Benaddi et al. [69], e Servin e Kudenko [70], negligenciaram aspectos práticos importantes, como a adaptação a mudanças no comportamento do tráfego de rede e a facilitação das atualizações do modelo em ambientes reais de produção. A proposta de Lopez-Martin et al. [47] tentou abordar essa questão, mas também falhou em considerar plenamente os desafios relacionados às atualizações de modelos em cenários dinâmicos de redes.

A pesquisa sobre Aprendizado Federado (FL) na detecção de intrusões em redes tem proporcionado avanços significativos, visando reduzir sobrecargas de comunicação e melhorar a eficiência do processo. No entanto, embora essas abordagens tenham se destacado em preservar a privacidade dos dados dos participantes, enfrentam desafios significativos na gestão do processo de atualização do modelo ao longo do tempo e na adaptação às mudanças no comportamento do tráfego de rede. Outros estudos, também apresentam soluções promissoras, assim, é imperativo que futuras pesquisas abordem essas lacunas para garantir a eficácia e a robustez dos modelos FL em ambientes dinâmicos de segurança de rede.

Capítulo 4

Avaliação do Estado da Arte

Esta seção investiga os principais aspectos que tornam a detecção de intrusões baseada em rede desafiadora para técnicas baseadas em aprendizado de máquina. Mais especificamente, realizamos essa tarefa propondo um novo conjunto de dados de intrusão e avaliando o desempenho de abordagens de detecção de intrusões amplamente utilizadas baseadas em aprendizado de máquina.

4.1. Conjunto de dados MAWIFlow

A confiabilidade dos mecanismos de detecção de intrusões em rede de computadores depende do uso de um conjunto de dados de treinamento realista. No entanto, as abordagens atuais na literatura frequentemente constroem suas técnicas com base em conjuntos de dados desatualizados com várias falhas conhecidas [42]. Trabalhos anteriores raramente são usados em ambientes de produção, apesar dos resultados promissores, como um modelo de ML de alta precisão [8], [9]. Na prática, um conjunto de dados de treinamento realista [14] deve conter tráfego de rede real que pode ser encontrado em ambientes de produção com a implementação válida e correta de protocolos de comunicação de tráfego de rede.

O tráfego de rede precisa ser altamente diverso em termos de protocolos utilizados e comportamento de rede, garantindo a disponibilidade completa dos protocolos de tráfego de rede no ambiente de produção, bem como sua variabilidade. Os eventos de rede devem ser previamente rotulados como pertencentes a uma classe (por exemplo, normal ou ataque) para possibilitar a avaliação adequada do modelo e a construção do classificador de ML. Finalmente, o conjunto de dados deve estar

publicamente disponível em um formato utilizável para permitir a adequada comparação de propostas de detecção de intrusões.

Normalmente, as propostas na literatura coletam tráfego de rede de um ambiente de teste controlado ou monitoram o comportamento do ambiente de produção para atender aos requisitos mencionados [12]. O primeiro permite o compartilhamento e a rotulagem adequada dos eventos devido à sua natureza controlada. No entanto, eles carecem de um conteúdo de rede altamente diverso do ambiente de produção e geram tráfego de rede irrealista. Em contraste, o último produz dados de rede reais e válidos, que são altamente diversos, mas dificultam a rotulagem adequada dos eventos e o compartilhamento público dos dados coletados (por exemplo, devido às preocupações de privacidade associadas ao tráfego de rede real). Independentemente da abordagem adotada para a construção dos conjuntos de dados, os trabalhos da literatura não consideraram adequadamente o comportamento evolutivo do tráfego de rede. Isso significa que mesmo os melhores conjuntos de dados falharão à medida que o tráfego de rede disponível permanecer inalterado nesses dados ao longo do tempo.

Nosso novo conjunto de dados, chamado MAWIFlow, compreende tráfego de rede real, válido e previamente rotulado, coletado de ambientes de produção por longos períodos. Mais especificamente, seus fluxos de rede são extraídos das trilhas de pacotes de rede MAWI [43] (SamplepointF no arquivo MAWI), que são coletadas diariamente com um intervalo de 15 minutos a partir de um link de trânsito de rede conectado entre o Japão e os EUA. Durante o período de gravação da rede, o ponto de amostragem era composto por um link de tráfego de rede de 1 Gbps. Além disso, os payloads dos pacotes de rede são removidos, e os campos sensíveis dos cabeçalhos dos pacotes de rede são anonimizados.

A rotulagem do tráfego de rede é uma tarefa desafiadora, que geralmente é realizada apenas com assistência humana [4]. Infelizmente, usar um administrador para a rotulagem dos eventos de rede torna-se inviável devido ao grande número de eventos de rede disponíveis em nosso conjunto de dados (≈ 7,23 bilhões). Em geral, a literatura realiza a tarefa de rotulagem por meio de técnicas baseadas em uso indevido (por exemplo, assinaturas de intrusão) [8], [12], o que pode introduzir falhas de rotulagem, tornando a tarefa de classificação fácil de ser realizada. O classificador de ML pode aprender o conjunto subjacente de assinaturas utilizadas, em vez do comportamento do tráfego de rede. Consequentemente, nosso conjunto de dados foi rotulado usando o trabalho do MAWILab [44], que rotula os eventos anômalos diários (fluxos de rede) do

MAWI utilizando uma variedade de detectores de anomalias não supervisionados. Embora tal procedimento de rotulagem seja propenso a erros, ele lida com a variabilidade do comportamento do tráfego de rede analisado, já que a tarefa de rotulagem é realizada diariamente. Isso mantém a variabilidade do comportamento do tráfego de rede e os desafios do tráfego de rede do ambiente de produção.

Tabela 4 – Estatísticas do conjunto de dados MAWIFlow

Característica	**Valor**
Média Diária de Pacotes de Rede	120 Milhões
Média Diária de Fluxos de Rede	23 Milhões
Média Diária de Taxa de Transferência	720 Mbps
Média Diária de Fluxos Anômalos	2.1 Milhões
Média Diária do Tamanho do Conjunto de Dados	23.2 GBs
Total de Pacotes de Rede	32.36 Bilhões
Total de Fluxos de Rede	7.23 TBs
Tamanho Total do Conjunto de Dados	8.3 TB

Neste trabalho, consideramos todo o tráfego de rede disponível para um intervalo de quatro anos, de 2016 a 2019. As anomalias de rede são classificadas de acordo com seus tipos de ataque, conforme rotuladas pelo MAWILab. Nesse caso, as anomalias de rede podem ser de vários tipos, incluindo varredura de serviço (Service Scan), varredura de TCP (TCP Scan) e negação de serviço (denial-of-service), entre outros ataques ao nível da rede.

Devido à sua grande escala, com mais de 8 TB de dados compostos por mais de 7 bilhões de fluxos de rede, o conjunto de dados MAWIFlow foi construído usando o algoritmo de extração de características BigFlow, que é implementado em um framework de processamento de Big Data. O algoritmo de extração de características extrai, para cada fluxo de rede, 39 características em um intervalo de janela de 15 segundos, conforme listado na Tabela 2. Cada fluxo de rede é previamente rotulado como normal ou ataque, conforme estabelecido pela saída dos algoritmos não supervisionados do MAWILab [44]. A Tabela 4 mostra as estatísticas do MAWIFlow ao longo do período avaliado de 4 anos.

O conjunto de dados construído fornece as propriedades esperadas de um conjunto de dados de tráfego de rede realista usado para a comparação de técnicas de detecção de intrusões [14]. O conjunto de dados fornece tráfego de rede real, válido e correto, pois foi coletado de um link de trânsito de rede real, mantendo as características dos ambientes de produção. O tráfego de rede disponível é altamente diverso, com um comportamento completo dos protocolos de comunicação devido ao período prolongado de gravação, considerando uma janela de tempo de 4 anos. Além disso, a usabilidade é garantida, pois o conjunto de dados é fornecido em formato de fluxo de rede em um ambiente publicamente disponível.

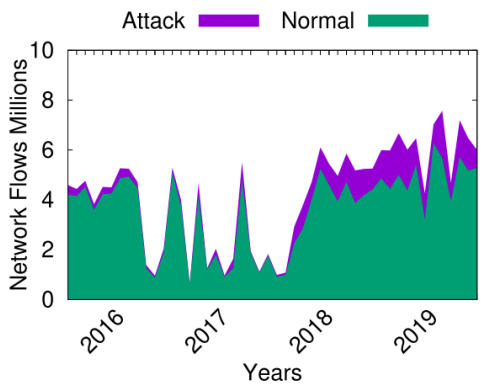
4.2. Confiabilidade do Aprendizado de Máquina para Detecção de Intrusões

A avaliação atual tem como objetivo responder às seguintes perguntas de pesquisa:

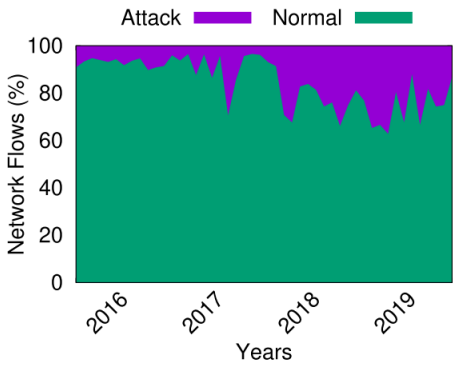
- (RQ1) Qual é o comportamento das abordagens amplamente utilizadas baseadas em aprendizado de máquina em termos de precisão ao longo do tempo quando nenhuma atualização de modelo é realizada?
- (RQ2) Qual é o impacto das atualizações periódicas de modelo na precisão das abordagens amplamente utilizadas baseadas em aprendizado de máquina?

A seguir, elaboramos mais sobre as avaliações realizadas e nossas descobertas.

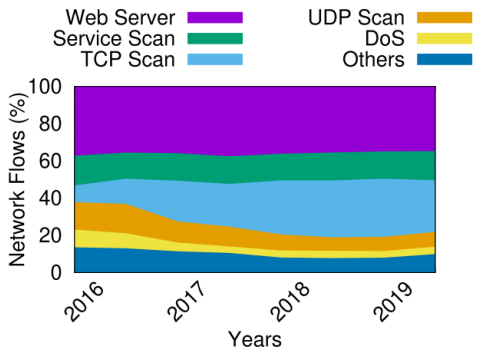
Para avaliar abordagens amplamente utilizadas baseadas em aprendizado de máquina, selecionamos quatro técnicas comumente usadas para detecção de intrusões, a saber, Long Short-Term Memory (LSTM), Random Forest (RF), Adaboosting (Ada) e Gradient Boosting (GBT).



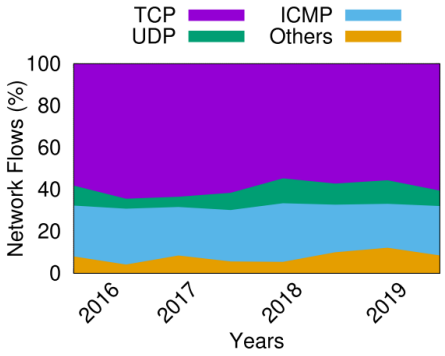
A) Número de fluxos de rede ao longo do tempo.



B) Distribuição dos fluxos de rede ao longo do tempo.



C) Distribuição do tráfego de rede de ataques ao longo do tempo.



D) Distribuição do tráfego de rede normal ao longo do tempo.

Figura 1 - Distribuição dos fluxos de rede do MAWIFlow ao longo dos 4 anos selecionados.

Os classificadores de conjunto (RF, Ada e GBT) foram implementados com 100 árvores de decisão como seus aprendizes-base, onde cada um deles também usa gini como métrica de qualidade de divisão de nó. O classificador GBT utiliza um valor de taxa de aprendizado de 0,1, com desvio como função de perda. O classificador Ada usa o SAMME como algoritmo de reforço e 1,0 como taxa de aprendizado. Os classificadores de conjunto foram implementados através da API do scikit-learn v0.24.

O LSTM foi implementado com a seguinte arquitetura:

- (I) Entrada: As 39 características de rede baseadas em fluxo são alimentadas como entrada para o LSTM;
- (II) LSTM: Duas camadas LSTM com 256 e 128 unidades, respectivamente;

- (III) Saída: Duas camadas densas implementadas com uma função de ativação relu, com 512 e 1 unidades, respectivamente. A arquitetura LSTM implementada usou a entropia cruzada categórica como perda usando o otimizador adam. Para o procedimento de construção do modelo, 1.000 épocas são executadas com um tamanho de lote de 512. É importante notar que os parâmetros usados no LSTM foram definidos a partir dos trabalhos relacionados, e não foram encontradas diferenças significativas ao variá-los.

Como mostrado na Figura 1(d), devido à natureza desbalanceada do MAWIFlow, uma subamostragem aleatória sem reposição é usada em cada procedimento de treinamento para equilibrar a ocorrência entre as classes para ambos os classificadores de conjunto e LSTM. Os classificadores foram avaliados quanto às suas taxas de falso negativo (FN), taxas de falso positivo (FP) e medida-F. As seguintes métricas de desempenho de classificação foram usadas:

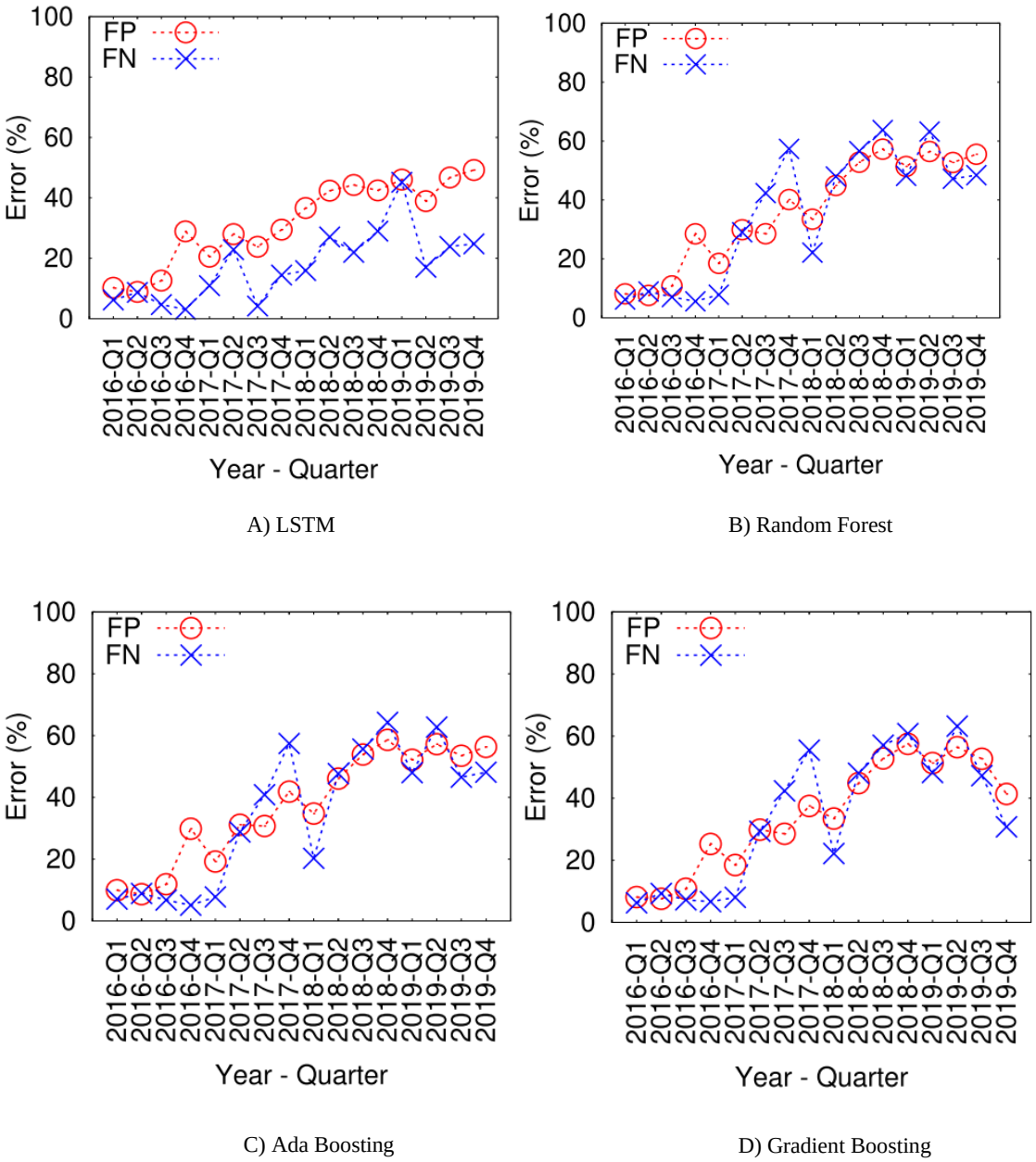
- Verdadeiro Positivo (TP): número de amostras de ataque corretamente classificadas como ataque.
- Verdadeiro Negativo (TN): número de amostras normais corretamente classificadas como normal.
- Falso Positivo (FP): número de amostras normais incorretamente classificadas como ataque.
- Falso Negativo (FN): número de amostras de ataque incorretamente classificadas como normal.

A medida-F foi calculada de acordo com a média harmônica dos valores de precisão e recall, considerando amostras de ataque como positivas e amostras normais como negativas.

4.3. Precisão de Classificação Sem Atualizações Periódicas do Modelo

O primeiro experimento visa responder à RQ1 e realiza a avaliação do desempenho da classificação das técnicas de ML selecionadas quando as atualizações do modelo não são realizadas ao longo do tempo. Os classificadores de ML selecionados são treinados apenas com os dados de janeiro de 2016, enquanto são usados para avaliação ao longo dos quatro anos restantes do intervalo. O objetivo da avaliação é medir como o

comportamento evolutivo do tráfego de rede afeta o desempenho da classificação ao longo do tempo se as atualizações do modelo não forem realizadas periodicamente, como comumente feito pelos trabalhos relacionados.



A) LSTM

B) Random Forest

C) Ada Boosting

D) Gradient Boosting

Figura 2 - Desempenho de precisão, mostrado trimestralmente, para vários algoritmos de ML em todo o conjunto de dados MAWIFlow. Os classificadores são treinados com dados de janeiro de 2016 e não são atualizados ao longo do tempo.

A Figura 2 mostra o erro médio de classificação para cada trimestre das técnicas de detecção de intrusões baseadas em ML sem atualizações periódicas do modelo. Um impacto significativo no desempenho da precisão pode ser notado em comparação ao seu

período de treinamento (ou seja, janeiro de 2016). Todas as abordagens de detecção de intrusões avaliadas diminuem sua precisão à medida que o tempo de vida do modelo aumenta. Por exemplo, em comparação com a fase de treinamento, o classificador RF aumentou, em média, 7,5% e 0,9% as taxas de FP e FN em 2016. Além disso, de 2017 a 2019, o classificador RF aumentou ainda mais suas taxas de FP e FN em comparação ao período de treinamento, apresentando um aumento médio de 38,3% e 35,3% nas suas taxas de FP e FN, respectivamente.

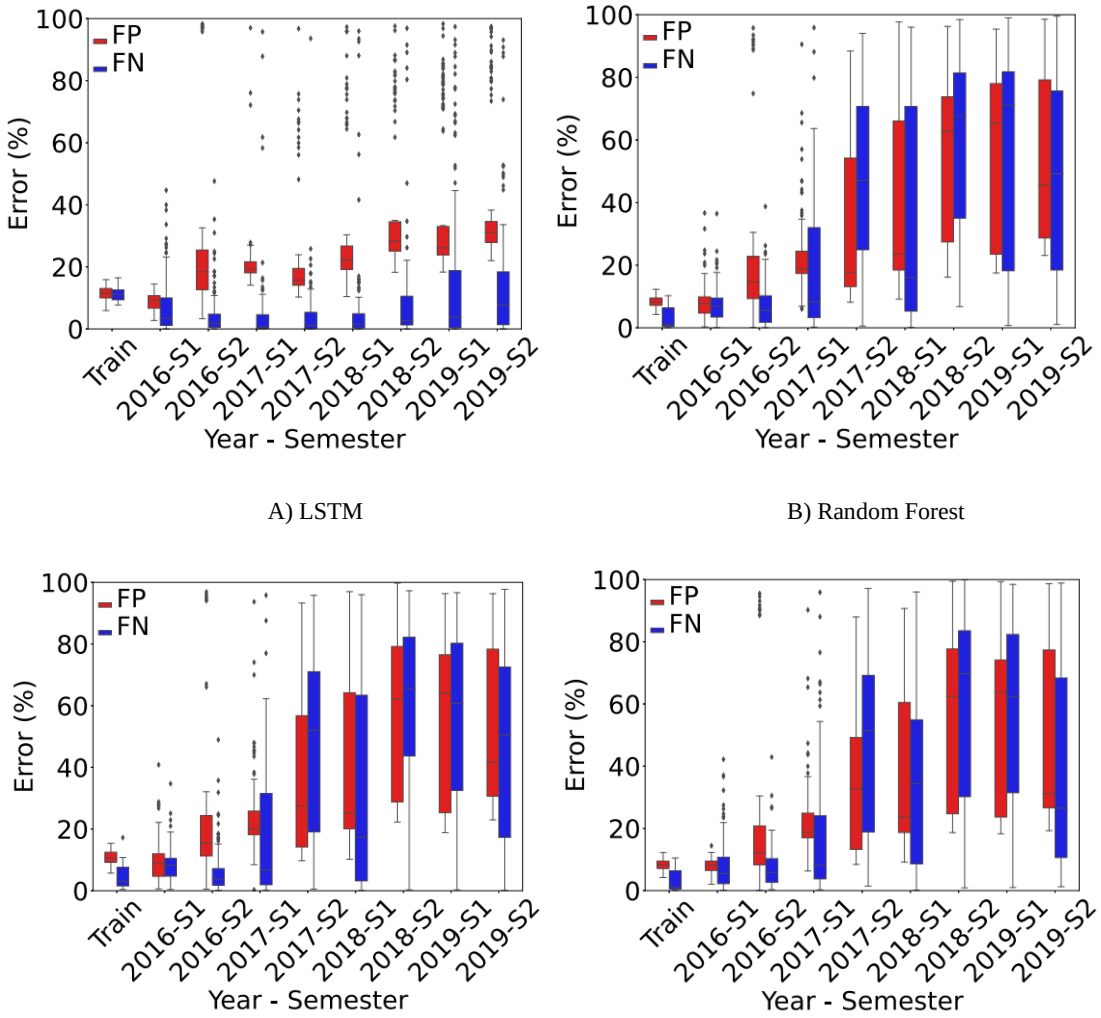
Em contraste, o classificador LSTM apresentou uma degradação de precisão menor com o passar do tempo. Por exemplo, alcançou uma taxa média de FN de apenas 13% em 2017. A redução na degradação da precisão do LSTM foi alcançada devido ao aumento da complexidade no modelo utilizado, dado que os classificadores rasos tradicionais (RF, Ada e GBT) não conseguem retratar todas as complexidades do tráfego de rede utilizado durante a fase de treinamento. Consequentemente, o modelo resultante atinge uma maior generalização em relação à classificação do tráfego de rede com o passar do tempo.

O desempenho das abordagens tradicionais de detecção de intrusões baseadas em ML depende do tempo de vida do modelo utilizado. A precisão média trimestral varia significativamente, de 2017-Q4 a 2018-Q1, para todos os classificadores avaliados. O comportamento não estacionário da precisão ao longo do tempo degrada significativamente a confiabilidade do mecanismo de detecção de intrusões, assumindo que o administrador descartará alertas sinalizados assim que maiores proporções de FP e FN forem experimentadas. Mesmo se um modelo desatualizado alcançar alta precisão de detecção (por exemplo, aqueles alcançados pelo classificador RF em 2018-Q1), os alertas do sistema serão descartados, pois sua precisão já degradou em circunstâncias anteriores.

A degradação da precisão dos classificadores selecionados é causada por mudanças no comportamento do tráfego de rede com o passar do tempo. Isso pode ser visto na Figura 1 (b), que mostra a distribuição da ocorrência de ataques no conjunto de dados MAWIFlow, onde a ocorrência de ataques permanece estável ao longo de 2016. Da mesma forma, as precisões de detecção do modelo também não são significativamente afetadas (veja a Figura 2Figura 1). Em contraste, a ocorrência de ataques muda significativamente em 2017, causada por um aumento significativo na varredura de TCP, seguido por uma diminuição nas varreduras de UDP e ataques de DoS, resultando na degradação da precisão das técnicas avaliadas. No entanto, a ocorrência de tráfego normal também muda com o passar do tempo, como mostrado na Figura 1(c).

Com o tempo, novos serviços de rede serão fornecidos e novos ataques de rede serão descobertos, mudando assim o comportamento subjacente da rede. No entanto, identificar tais mudanças no tráfego de rede subjacente (que afetam o modelo de ML implantado) é difícil, pois o administrador deve (frequentemente de forma manual) avaliar a precisão atual do modelo, conforme definido pela proporção de amostras de rede corretamente classificadas. Em contraste, o rótulo adequado do evento não está disponível em configurações de produção, tornando a avaliação da precisão significativamente mais desafiadora. Como resultado, o modelo de ML implantado deve ser regularmente atualizado para lidar com essas mudanças no tráfego de rede.

Para investigar mais a variabilidade da precisão ao longo do tempo quando as atualizações do modelo não são realizadas, comparamos a distribuição das precisões obtidas no treinamento (janeiro de 2016) com as obtidas no período restante. A Figura 3 mostra a distribuição das precisões (boxplot) dos classificadores avaliados sem atualizações do modelo ao longo do tempo.



A) LSTM

B) Random Forest

C) Ada Boosting

D) Gradient Boosting

Figura 3 - Distribuição da precisão, mostrada semestralmente, para vários algoritmos de ML em todo o conjunto de dados MAWIFlow. Os classificadores são treinados com dados de janeiro de 2016 e não são atualizados ao longo do tempo.

Independentemente da técnica de detecção, os intervalos de erro aumentam significativamente com o passar do tempo em comparação aos medidos durante o período de treinamento. Por exemplo, em média, o intervalo interquartil de erro do RF aumenta em 3,4% e 4,0% para taxas de FP e FN a cada seis meses de vida adicional do modelo, aumentando assim significativamente sua variação de precisão ao longo do tempo. As técnicas avaliadas não podem fornecer confiabilidade na detecção de intrusões meses após a fase de treinamento do classificador do modelo. No entanto, a distribuição da precisão aumenta com o passar do tempo, como observado pelo aumento no número de outliers (Figura 3, 2016 e 2017) e também nos intervalos interquartis ao longo do tempo (Figura 3, 2018 e 2019).

Pode-se notar que o comportamento evolutivo do tráfego de rede, que varia com o passar do tempo, representa um desafio significativo para as técnicas baseadas em ML.

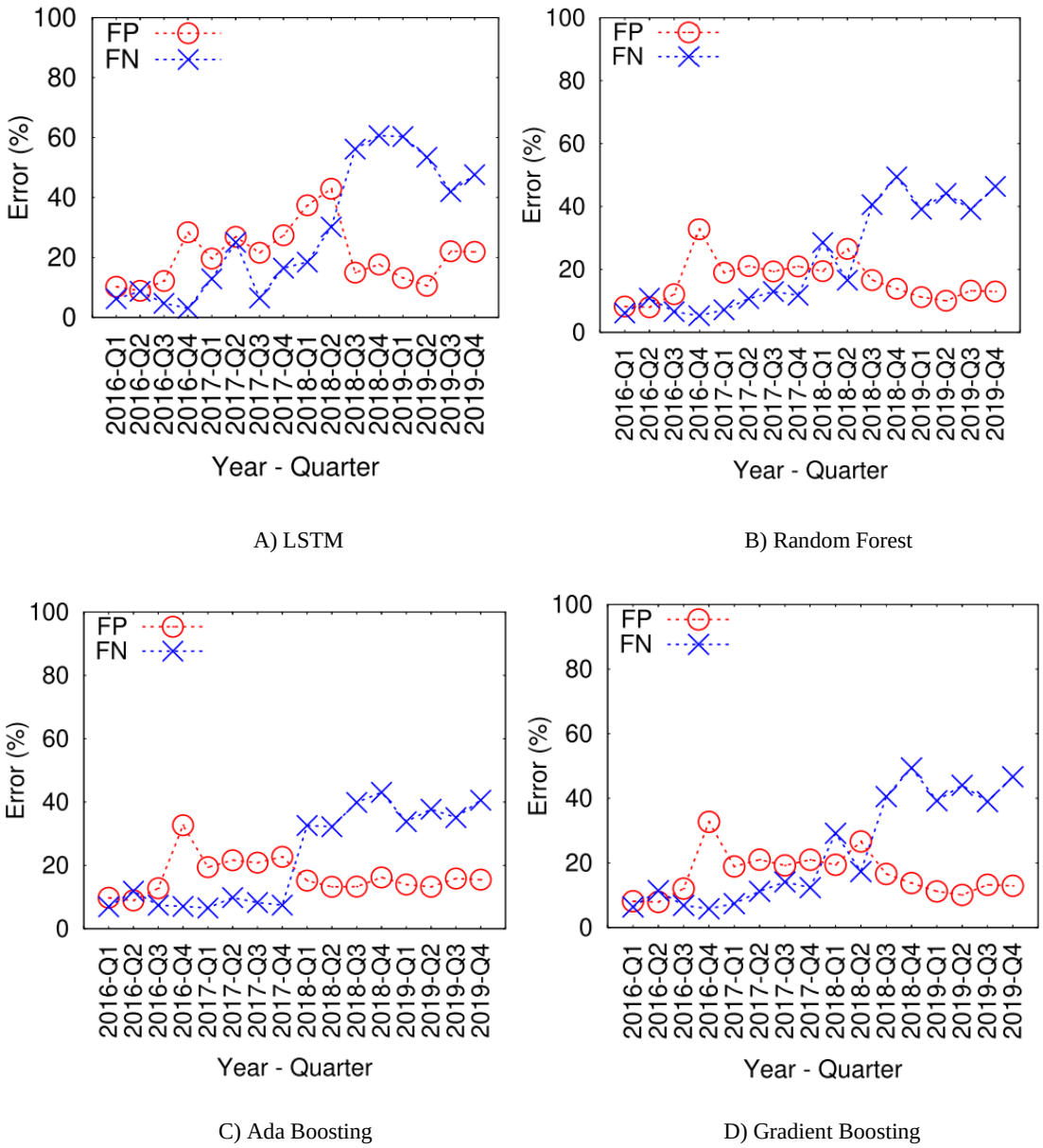
Torna-se inviável para o administrador de rede gerar um conjunto de dados de treinamento com todas as possíveis variações. Assim, o modelo de ML implantado apresentará uma alta variação de precisão se não for projetado para considerar as capacidades de generalização. Deve-se notar que uma alta variação na taxa de erro ao longo do tempo para o mecanismo de detecção de intrusões afeta significativamente a percepção do administrador do sistema sobre a confiabilidade do sistema.

Essa falta de confiabilidade ocorre porque, mesmo que o mecanismo de detecção de intrusões possa alcançar altas taxas de precisão com um modelo de ML desatualizado, o administrador suprimirá os alertas assim que uma alta taxa de erro for experimentada. Os mecanismos de detecção de intrusões devem alcançar altas taxas de precisão e apresentar baixa variabilidade de precisão ao longo do tempo para uma implantação confiável em produção. No entanto, se o modelo de ML não for atualizado, os classificadores de ML avaliados não serão capazes de lidar com o comportamento evolutivo do tráfego de rede.

As técnicas se tornam não confiáveis nos meses imediatos após o período de treinamento, como observado por taxas de erro mais altas (Figura 2) e maior variabilidade nas taxas de erro medidas (Figura 3). Portanto, as abordagens de detecção de intrusões baseadas em ML devem ser regularmente atualizadas para manter a confiabilidade da classificação.

4.4. Precisão de Classificação com Atualizações do Modelo

Para responder à RQ2, investigamos o impacto que as atualizações periódicas do modelo causam na precisão da classificação do modelo de ML subjacente. Primeiro, consideramos uma durabilidade do modelo de 6 meses usando um intervalo de 1 mês como dados de treinamento, o que significa que atualizamos o modelo de ML a cada semestre usando os dados do MAWIFlow que ocorreram um mês antes. Os classificadores avaliados são atualizados no final de janeiro e julho de cada ano - usando os dados que ocorreram nos últimos 30 dias daquele período. No entanto, é essencial mencionar que a frequência das atualizações do modelo deve ser estabelecida de acordo com as necessidades do administrador.



A) LSTM

B) Random Forest

C) Ada Boosting

D) Gradient Boosting

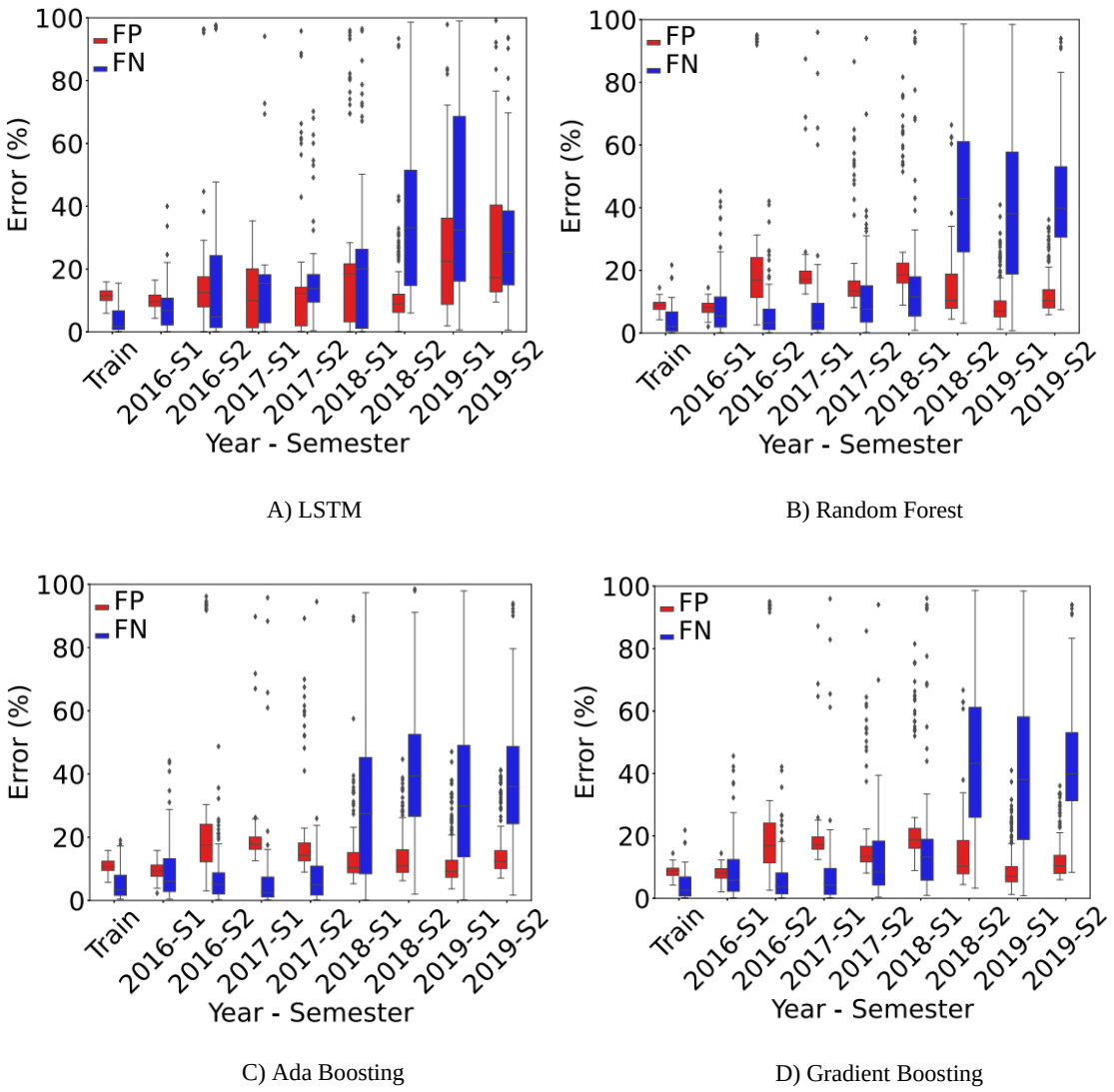
Figura 4 - Desempenho de precisão, mostrado trimestralmente, para vários algoritmos de ML em todo o conjunto de dados MAWIFlow. Os classificadores são atualizados a cada 6 meses, com 1 mês de dados de treinamento.

A Figura 4 mostra o desempenho da precisão ao longo dos dados do MAWIFlow quando as atualizações do modelo são realizadas a cada 6 meses. Pode-se notar que, em geral, o desempenho de precisão dos classificadores avaliados é melhorado. Por exemplo, o classificador RF atualizado periodicamente melhorou sua taxa média de FP de 35,1% para 16,5%, enquanto melhorou sua taxa média de FN de 36,0% para 23,4% em comparação com os resultados sem atualizações do modelo. A durabilidade do modelo varia ao longo do tempo, conforme observado por taxas mais altas de FP e FN em 2018 e 2019. O LSTM atualizado não conseguiu melhorar significativamente as taxas de precisão obtidas em comparação com os classificadores rasos. Pelo contrário, diminuiu as taxas médias de precisão assim que uma maior proporção de ataques é evidenciada

(Figura 1(a), 2018 e 2019). Essa diminuição ocorre porque, à medida que a complexidade do modelo LSTM aumenta em comparação com os classificadores rasos tradicionais, um número maior de amostras de treinamento deve ser fornecido para garantir a generalização adequada do modelo, como ocorreu com seu equivalente sem atualização.

A avaliação dos classificadores deveria ter sido atualizada com mais frequência para aumentar a confiabilidade nesse caso. Devido às dificuldades relacionadas à identificação da obsolescência do modelo, o administrador aumentará a periodicidade das atualizações do modelo, independentemente de o modelo atual ainda ser confiável, como ocorreu de 2016 a 2017. O aumento repentino nos ataques de rede mostra a necessidade de uma atualização mais frequente do modelo. Como mostrado na Figura 1(a), a ocorrência de ataques aumentou de ≈ 6% ao longo de 2016 e 2017 para ≈ 24% em 2018 e 2019.

As mudanças na ocorrência de ataques de rede afetam significativamente a precisão da classificação dos classificadores atualizados periodicamente, conforme observado pela degradação do desempenho da classificação de ataques em comparação com o desempenho da classificação normal (Figura 4, o aumento significativo em FN em comparação com FP). À medida que a ocorrência de ataques aumenta, também aumenta a complexidade de detectar ataques na rede, exigindo atualizações mais frequentes do modelo de ML. Os classificadores atualizados podem fornecer alta precisão de detecção para ameaças previamente conhecidas, considerando que ataques mais antigos ainda são frequentes com o passar do tempo (ver Figura 1(b)).



A) LSTM

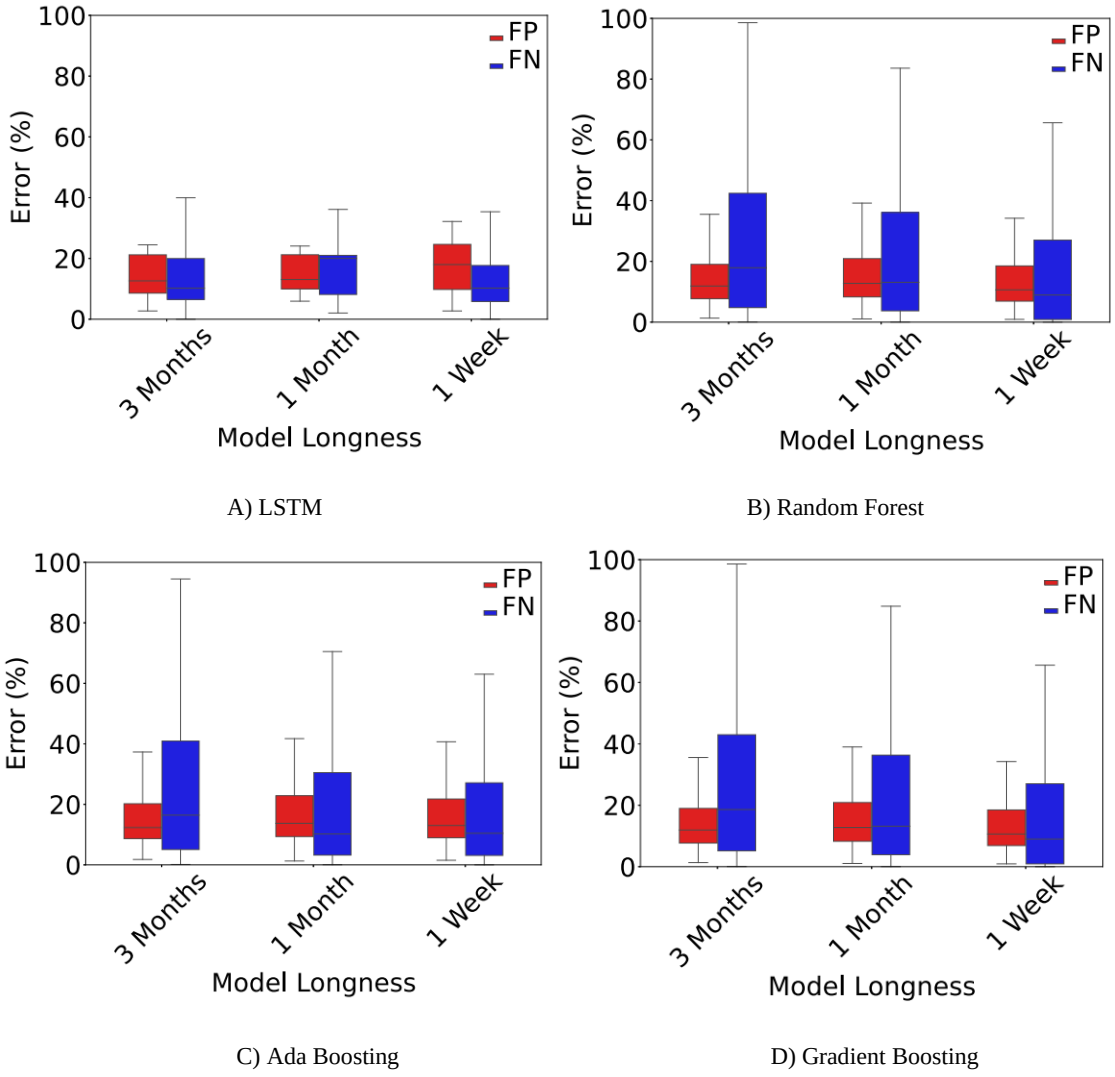
B) Random Forest

C) Ada Boosting

D) Gradient Boosting

Figura 5 - Distribuição da precisão, mostrada semestralmente, para vários algoritmos de ML em todo o conjunto de dados MAWIFlow. Os classificadores são atualizados a cada 6 meses, com 1 mês de dados de treinamento.

Também investigamos a variabilidade da precisão quando as atualizações do modelo são realizadas. A Figura 5 mostra a distribuição da precisão (boxplot) dos classificadores de ML avaliados quando as atualizações do modelo são realizadas a cada semestre. Em contraste com a abordagem sem atualizações do modelo (Figura 3), as atualizações periódicas do modelo também diminuem a variação da precisão ao longo do tempo, conforme observado por uma faixa interquartil menor tanto em 2016 quanto em 2017. Notavelmente, em 2018 e 2019 (o período que exige mais atualizações frequentes do modelo), a faixa interquartil aumenta até seu equivalente sem atualizações periódicas. Mesmo considerando uma durabilidade de modelo de 6 meses, um modelo de ML desatualizado pode diminuir e variar significativamente sua precisão, impactando a confiabilidade da detecção de intrusões.



A) LSTM

B) Random Forest

C) Ada Boosting

D) Gradient Boosting

Figura 6 - Troca entre a duração do modelo e a taxa média de erro no MAWIFlow. A duração do modelo estabelece a frequência de atualização do modelo, enquanto a taxa média de erro é medida como a média das taxas de FP e FN ao longo de todo o conjunto de dados MAWIFlow.

Investigamos ainda como a periodicidade da atualização do modelo pode afetar o desempenho da classificação dos classificadores de ML selecionados. A Figura 6 mostra as precisões do modelo de acordo com a durabilidade do modelo, ou seja, a frequência com que as atualizações do modelo são realizadas. É possível notar que aumentar a periodicidade da atualização do modelo, na maioria dos casos, melhora as precisões do modelo obtido. Por exemplo, a faixa interquartil de FN do RF atualizado semanalmente diminuiu 38% em comparação com seu equivalente atualizado trimestralmente. O classificador LSTM não é tão significativamente afetado pelo aumento da periodicidade da atualização do modelo. Mais especificamente, a

durabilidade de modelo de 1 mês do LSTM diminuiu a faixa interquartil de FN em apenas 4,1% em comparação com seu equivalente de durabilidade de modelo de 3 meses. Isso ocorre porque, conforme discutido anteriormente, a maior complexidade do LSTM, em comparação com os classificadores rasos, pode aumentar a durabilidade do modelo.

4.5. Discussão

Em geral, os trabalhos anteriores assumem um comportamento de tráfego de rede estático, utilizando conjuntos de dados que não representam a natureza dinâmica dos ambientes de produção, resultando em proposta de detecção imprecisos e irreais. Ao contrário dos existentes na literatura, nosso conjunto de dados MAWIFlow é o primeiro do tipo e é composto por mais de 8TB de dados que abrangem quatro anos.

O conjunto de dados nos permitiu avaliar as abordagens atuais na literatura quanto à sua confiabilidade ao longo do tempo. Os experimentos mostraram que as técnicas amplamente usadas baseadas em ML não conseguem lidar com o comportamento evolutivo do tráfego de rede. Com o passar do tempo, sua precisão de detecção diminui enquanto a variação da precisão aumenta.

A execução de atualizações periódicas do modelo pode impactar positivamente a precisão e diminuir sua variação. No entanto, isso deve ser feito de acordo com a distribuição do tráfego de rede. As abordagens da literatura atual devem ser usadas com uma durabilidade de modelo pequena (por exemplo, alguns dias ou semanas) para fornecer confiabilidade ao longo do tempo. Devido à expectativa de pequena durabilidade do modelo, as técnicas propostas anteriormente raramente são usadas em ambientes de produção. A principal razão é que o modelo de ML com uma vida útil tão curta não pode ser atualizado com a frequência necessária para garantir a confiabilidade do sistema.

Capítulo 5

Proposta

Esta seção descreve os dois modelos propostos de detecção de intrusão de rede confiável. Enquanto uma proposta aborda o comportamento em evolução do tráfego de rede ao longo do tempo, conduzindo a classificação com uma opção de rejeição enquanto aproveita atualizações do modelo implementadas seguindo uma abordagem de FL, a outra abordagem é um novo modelo confiável de aprendizado por reforço capaz lidar com a variabilidade do comportamento de rede.

O objetivo principal das propostas é manter a confiabilidade na classificação do sistema por períodos mais longos sem atualizações do modelo. Além disso, a proposta busca reduzir os recursos humanos e computacionais necessários para realizar essa tarefa quando as atualizações do modelo forem realizadas.

5.1. Proposta Aprendizado por Reforço

Nesta seção é apresentado um novo modelo confiável de aprendizado por reforço para detecção de intrusões para abordar o comportamento evolutivo do tráfego de rede mencionado anteriormente. O objetivo principal é manter a confiabilidade na classificação do sistema por períodos mais longos sem atualizações do modelo. Além disso, a proposta visa reduzir os recursos humanos e computacionais necessários para realizar essa tarefa quando as atualizações do modelo forem realizadas. A Figura 7 mostra uma visão geral do modelo proposto, que está organizado em duas etapas principais: Classificação Confiável e Construção de Agente Confiável.

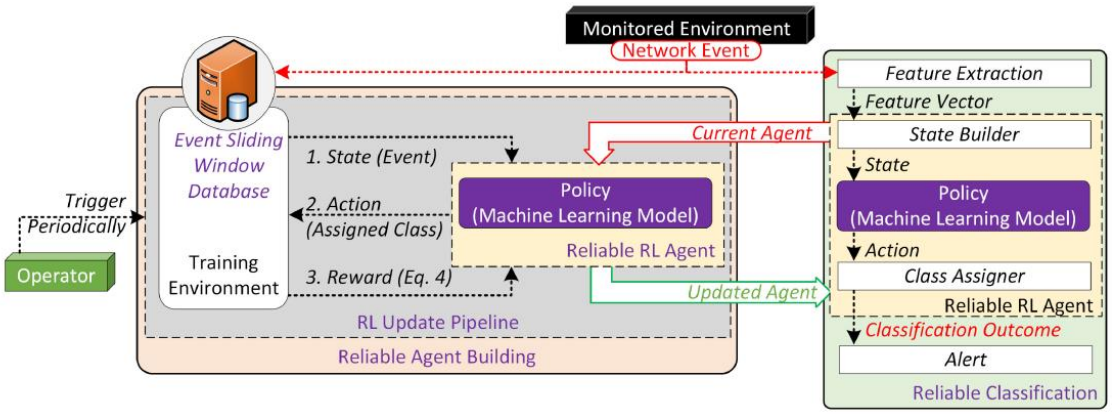


Figura 7 - Visão geral da classificação e das pipelines de atualização do modelo de detecção de intrusões baseado em aprendizado por reforço confiável.

A Classificação Confiável é realizada através de um pipeline de classificação por meio de aprendizado por reforço. Seu objetivo é classificar eventos de rede ao longo do tempo em implantações de produção, seja como normal ou ataque. Os eventos de rede são representados como um estado do ambiente, enquanto o agente de aprendizado por reforço gera uma ação representando a classe do evento classificado de acordo com a política de RL conhecida.

O objetivo da Construção de Agente Confiável é duplo: a construção da abordagem proposta na implantação inicial e a atualização de um agente desatualizado já implantado. A primeira implantação do agente é executada em uma abordagem de aprendizado por reforço com um agente construído do zero. A atualização do modelo diminui os custos computacionais aproveitando o agente desatualizado atual implantado em produção, seguindo uma lógica de aprendizado por *transfer learning*. Nesse caso, um conjunto reduzido de eventos pode ser usado, aproveitando o conhecimento prévio do agente desatualizado atual, aplicando assim um mecanismo de janela deslizante de eventos ao ambiente de produção - por exemplo, atualizando o agente com eventos da última semana.

A proposta pode construir políticas de agente através de um novo algoritmo de aprendizado por reforço que estende a durabilidade do modelo obtido. A proposta, tem objetivo de aumentar a durabilidade do modelo através da correção do modelo, em vez de apenas focar na precisão do modelo. A proposta representa a diferença de correção do modelo entre a confiança da classificação de saída do modelo e o rótulo correto do evento. Como resultado, o treinamento do modelo visa uma maior precisão e aproximação ao rótulo do evento, considerando sua distância para a classe correta do evento. Nas

subseções seguintes, será detalhado a proposta, apresentada a arquitetura dos módulos que a implementam e descrevemos seus principais componentes.

5.1.1. Construção de agente confiável

O objetivo principal do modelo proposto é aproveitar a técnica de aprendizado por reforço para construir uma política de agente com maior durabilidade em produção, ou seja, apresentando uma vida útil mais longa. A abordagem de Construção de Agente Confiável trata a detecção de intrusões como uma tarefa de aprendizado por reforço, fazendo uso das seguintes relações:

- **Ambiente**: O conjunto de dados de treinamento de detecção de intrusões que contém um conjunto de eventos de rede esperados para ocorrer na implantação em produção.
- **Estado**: Evento atual do ambiente que o agente de detecção de intrusões deve analisar. Pode ser composto por um evento normal ou de ataque.
- **Ação**: O resultado da classificação do agente de um determinado estado do ambiente, seja como normal ou ataque.
- **Recompensa**: A recompensa que um agente recebe por uma determinada ação de acordo com o estado atual do ambiente.

A lógica da nossa abordagem de detecção de intrusões como uma tarefa de aprendizado por reforço é mostrada na Figura 7 (Construção de Agente Confiável). Pode-se notar que um agente recebe um estado do ambiente de conjunto de dados de treinamento (ou seja, um evento) como entrada. Ele age usando uma classe atribuída por evento, recebendo a recompensa correspondente. O procedimento de treinamento pode ser executado usando um novo agente (por exemplo, o modelo de primeira implantação) ou um desatualizado (por exemplo, uma atualização de modelo). O operador é responsável por acionar periodicamente o processo de construção de agente confiável (por exemplo, a cada semestre).

Deve-se notar que a periodicidade da atualização do modelo deve ser definida de acordo com as necessidades do administrador. A proposta se baseia em uma janela deslizante de eventos para facilitar as atualizações do modelo nesse caso. Assume-se que um número menor de eventos pode ser usado nas atualizações do modelo se o conhecimento prévio do comportamento histórico do ambiente estiver disponível. Portanto, aproveita o conhecimento do agente desatualizado para representar o comportamento histórico do ambiente enquanto é atualizado por meio de uma lógica de *transfer learning*.

Como resultado, as atualizações do modelo podem ser realizadas como um simples ajuste fino da política do agente em vez de construí-la novamente do zero. Uma atualização de modelo que pode ser realizada com menos eventos exigirá menos intervenção humana para o procedimento de rotulagem e menores custos computacionais.

A construção de um novo conjunto de dados de treinamento no modelo é realizada através de uma janela deslizante de eventos do ambiente de produção. Nesse caso, os dados da rede que ocorreram N dias antes da tarefa de atualização do modelo devem ser armazenados e devidamente rotulados (Figura 7, Banco de Dados da Janela Deslizante de Eventos). O administrador pode usar técnicas de ML não supervisionadas, abordagens de detecção baseada em uso indevido ou até mesmo armazenar os dados da rede por períodos prolongados até a devida divulgação do ataque em domínios públicos. Mesmo que o procedimento de atualização do modelo exija um período prolongado para sua execução, o modelo proposto manterá sua confiabilidade, pois também é projetado para aumentar a durabilidade do modelo, ou seja, sua vida útil. Além da facilidade de atualizações do modelo, o desafio da durabilidade do modelo também deve ser abordado.

A proposta usa as métricas de correção de classificação como recompensas na fase de treinamento para estender a durabilidade do modelo de ML. O cálculo da recompensa do agente é mostrado na Equação 2, onde $confiança^{inst}$ representa a confiança da classificação do modelo de ML subjacente em uma determinada $confiança^{inst}$, FP denota falsos positivos e FN denota falsos negativos.

$$recompensa = \begin{cases} 1 - confiança^{inst}, & if\ FP\ ou\ FN \\ confiança^{inst}, & caso\ contrário \end{cases}$$

Equação 2

Classificações confiantes em eventos classificados incorretamente recebem menos recompensas, enquanto eventos corretamente classificados recebem sua confiança

de classificação como recompensa. Assim, as recompensas são calculadas como uma medida de aproximação da confiança do classificador em relação à classe correta do evento. Os valores de confiança do classificador são independentes do algoritmo de classificador utilizado. Por exemplo, o classificador *Random Forest* fornece seus valores de confiança como a proporção de suas árvores de decisão base que classificam uma determinada instância como a classe atribuída ao evento.

O objetivo da proposta é aumentar a durabilidade do modelo enquanto diminuí a variação da precisão usando a correção da classificação, medida através dos valores de confiança do classificador, em vez de apenas buscar uma maior precisão. Como resultado, o modelo de ML construído fornecerá melhor correção de classificação em todos os eventos de entrada, em vez de aumentar a precisão em um subconjunto (resultando em menor longevidade causada pelo sobreajuste no modelo de dados de treinamento).

Para implementar o novo modelo de aprendizado por reforço para detecção de intrusões, usando a medida de recompensa proposta, foi desenvolvido uma versão do conhecido algoritmo de aprendizado por reforço Q-Learning [45]. A implementação, executada em cada rodada de treinamento ou atualização do modelo (Figura 7, Construção de Agente Confiável), é mostrada no Algoritmo 1.

Algorithm 1 Proposed Intrusion Detection Q-Learning Algorithm for Reliable Reinforcement Learning Agent

Require:
States $S = \{instance_1, \ldots, instance_n\}$ ▷ Training dataset
Actions $A = \{normal, attack\}$ ▷ ML NIDS Classes
Reward function $R{:}S \times A \to \mathbb{R}$ ▷ Reward function as computed through Eq. (4)
Transition function $R : S \times A \to \mathbb{S}$
Learning rate $\alpha \in [0, 1]$, typically $\alpha = 0.1$
Discounting factor $\gamma \in [0, 1]$
procedure QLEARNING(S, A, R, T, α, γ)
  Initialize $Q : S \times A \to \mathbb{R}$ arbitrarily, or outdated agent
  while Q is not converged **do**
    Start in random state $s \in S$
    while s is not terminal **do**
      Calculate π according to Q and exploration strategy (e.g. $\pi(x) \leftarrow arg\ max_a\, Q(x, a)$)
      $a \leftarrow \pi(s)$ ▷ Establishes state classification confidence
      $r \leftarrow RewardFunction(s, a)$ ▷ Receive the reward as computed through Eq. (4)
      $s' \leftarrow GetInstance(instance_i + 1)$ ▷ Receive the new state
      $Q(s', a) \leftarrow (1 - \alpha) \cdot Q(s, a) + \alpha \cdot (r + \gamma \cdot \max_{a'} Q(s', a'))$
      $s \leftarrow s'$
    end while
  end while
  return Q
end procedure

Algoritmo 1 - Algoritmo de Q-Learning para Agente Confiável de Aprendizado por Reforço

O algoritmo recebe um conjunto de dados de treinamento, semelhante ao algoritmo Q-Learning tradicional, que atua como os estados do ambiente (S). Ele inicializa um agente de forma arbitrária, na primeira implantação do modelo, ou com um agente desatualizado, em um processo de atualização de modelo que segue a lógica de *transfer learning*. O algoritmo é executado até que ele tenha convergido, atingindo, por exemplo, um nível de precisão esperado ou um número predefinido de iterações. Em resumo, ele seleciona aleatoriamente a cada iteração um estado (ou seja, uma instância) do conjunto de dados de treinamento (ou seja, um ambiente), gera uma ação relacionada (ou seja, confiança) e recebe uma recompensa de acordo (Eq. (2)). Assim, aproxima os valores de confiança do modelo de classificação ao rótulo correto de cada evento para maximizar suas recompensas ao longo do tempo, melhorando sua capacidade de generalização sobre todo o conjunto de dados de treinamento (ambiente).

5.1.2. Confiança da Classificação

A tarefa de classificação em implantações de produção é mostrada na Figura 7, (Classificação Confiável). Ele recebe um evento de tráfego de rede como entrada do ambiente monitorado (por exemplo, um pacote de rede). Os dados coletados são então representados como um vetor de características, extraído por um módulo de extração de características (por exemplo, extração de várias características baseadas em fluxo, conforme mostrado na Tabela 2). O vetor de características é alimentado como entrada para o agente de aprendizado por reforço confiável implantado em produção. O agente então representa o vetor de características como o estado do ambiente (Figura 7, Construtor de Estado) e aplica a política do agente com seu modelo de ML subjacente. A política gera a ação (valores de confiança) e o resultado da classificação pode ser estabelecido (Figura 7, Atribuidor de Classe). Finalmente, um alerta pode ser gerado se o evento for classificado como um ataque.

5.1.3. Discussão

O comportamento evolutivo do tráfego de rede é um desafio para a detecção de intrusões baseada em ML. A proposta visa abordar três aspectos principais relacionados às mudanças no comportamento do tráfego de rede ao longo do tempo: as atualizações do modelo, a durabilidade do modelo e a variação da precisão. A proposta aproveita o

conhecimento prévio sobre o ambiente para facilitar as atualizações do modelo usando uma lógica de *transfer learning*. A suposição é que o agente desatualizado pode ser usado na tarefa de atualização do modelo para diminuir os custos computacionais, a quantidade de dados de treinamento necessários e os custos associados à tarefa de rotulagem de eventos. A correção do modelo proposto busca uma maior durabilidade do modelo. Desta forma, se deve buscar a aproximação do modelo a todos os eventos de treinamento para aumentar a durabilidade do modelo. Portanto, a proposta não favorece apenas um subconjunto de eventos, como comumente feito na literatura por abordagens baseadas em precisão. A variação da precisão é diminuída pela durabilidade do modelo - treinado para buscar maior correção do modelo em todos os dados de treinamento.

5.2. Proposta Aprendizado Federado

A Figura 8 mostra uma visão geral do modelo proposto, que está organizado em duas etapas principais: Classificação Confiável e Construção de Agente Confiável. O principal objetivo da proposta é aliviar a carga do operador de rede para atualizar o modelo de detecção de intrusão implantado, ao mesmo tempo em que mantém o sistema confiável. O modelo proposto é implementado em duas fases, a Classificação Confiável e a Atualização de Modelo Baseada em FL, como mostrado na Figura 8.

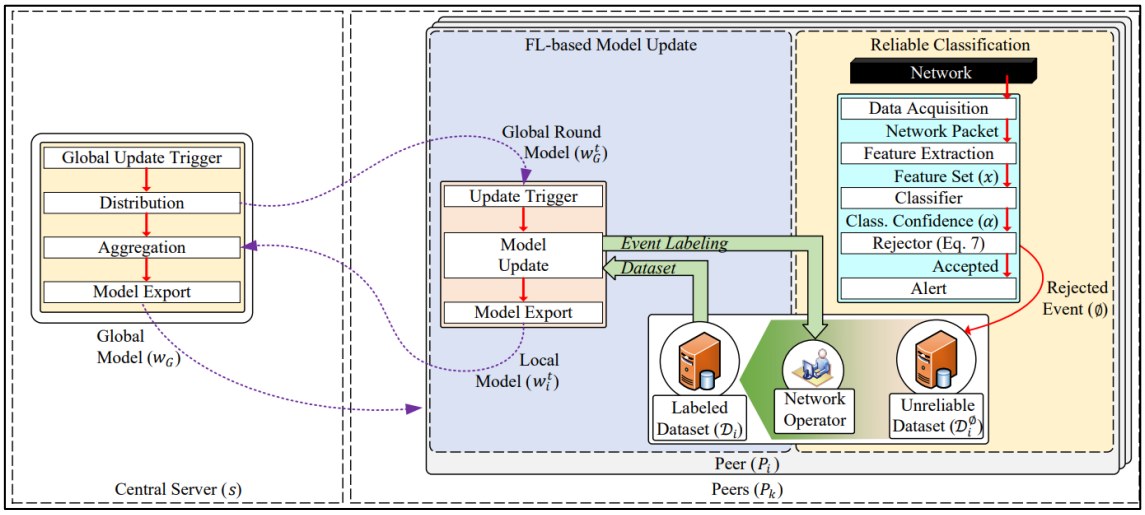


Figura 8 - Modelo proposto de aprendizado federado para atualizações confiáveis de modelos em detecção de intrusão em redes.

O principal objetivo da proposta é aliviar a carga do operador de rede para atualizar o modelo de detecção de intrusão implantado, ao mesmo tempo em que mantém o sistema confiável. O modelo proposto é implementado em duas fases, a saber, Classificação Confiável e Atualização de Modelo Baseada em FL, como mostrado na Figura 8.

A proposta considera uma arquitetura de FL composta por um servidor central e um conjunto de peers $\boldsymbol{P_k}$, onde $\boldsymbol{k}$ denota o número de peers, com $\boldsymbol{k > 1}$. Cada peer $\boldsymbol{P_i}$ é uma organização que visa realizar uma detecção confiável de intrusão, facilitando o processo de atualização do modelo. O servidor central $\boldsymbol{s}$ é confiável para todos os peers. O servidor central é responsável por compor o modelo global $\boldsymbol{W_G}$, executando as tarefas de inicialização, distribuição e agregação. Por outro lado, cada peer FL $\boldsymbol{P_i}$ executa a tarefa de treinamento local com base em seu conjunto de dados rotulados privados $\boldsymbol{D_i}$.

Cada peer executa o módulo de Classificação Confiável (Figura 8) para realizar a detecção de intrusão de rede em sua infraestrutura. A proposta visa alcançar confiabilidade na classificação, sendo realizada com uma opção de rejeição. O principal objetivo é identificar classificações pouco confiáveis com base nos valores de confiança produzidos pelos classificadores implantados à medida que o tempo passa. Vale ressaltar que a confiança na classificação é independente do classificador. Por exemplo, o classificador RF geralmente produz seus valores de confiança de classificação de acordo com a proporção de árvores de decisão individuais que classificam o evento de entrada para um determinado rótulo. A proposta assume que as classificações de baixa confiança têm uma maior probabilidade de produzir um falso positivo, portanto, elas devem ser rejeitadas pelo sistema. Eventos rejeitados são usados de duas maneiras para a validação dos alertas e atualizações do modelo. Eventos rejeitados, devido à sua baixa confiança na classificação, não geram alertas para o operador de rede, pois têm maior probabilidade de produzir um alerta falso. No entanto, assume-se que os eventos rejeitados representam novos comportamentos desconhecidos pelo modelo de ML implantado, portanto, eles são armazenados para as próximas atualizações do modelo. Como resultado, a abordagem pode manter sua confiabilidade de classificação à medida que o tempo passa, selecionando proativamente quais eventos devem ser usados para atualizações do modelo.

Para lidar com o comportamento não estacionário do tráfego de rede, a proposta utiliza o módulo de Atualização de Modelo Baseada em FL, que visa facilitar a atualização do modelo de duas maneiras. Primeiro, à medida que o tempo passa, cada peer $\boldsymbol{P_i}$ realiza atualizações periódicas do modelo por meio de uma abordagem de FL,

atualizando o modelo global W^t_G a cada rodada de execução t para construir conjuntamente um modelo global W_G. Como resultado, cada organização participante pode diminuir os custos relacionados à tarefa de atualização do modelo compartilhando seu conhecimento CTI sem afetar sua privacidade. Os dados de treinamento necessários para conduzir as atualizações do modelo podem ser significativamente reduzidos à medida que as organizações compartilham o conhecimento obtido a partir de seus dados locais. Em segundo lugar, é conduzido as atualizações periódicas do modelo por meio de uma abordagem de *transfer learning*, utilizando a abordagem de FL para os eventos rejeitados $D^{\emptyset}_i$ de cada peer. Assim, o modelo proposto pode reduzir significativamente os custos computacionais durante as atualizações do modelo, pois cada peer colaborativo reconstrói o modelo global desatualizado. Essas suposições reduzirão os dados necessários para realizar essa tarefa, tornando a tarefa de atualização do modelo mais fácil para o operador de rede.

As próximas subseções descrevem mais detalhadamente o modelo RF proposto e os detalhes de implementação.

5.2.1. Confiança da Classificação

O comportamento do tráfego de rede do mundo real é altamente variável à medida que o tempo passa. Esse fenômeno afeta a confiabilidade das abordagens projetados de sistemas de detecção de intrusão baseados em aprendizado de máquina, aumentando a taxa de erro medida em comparação com o período de treinamento (ver Figura 8). Para enfrentar esse desafio, o operador de rede deve realizar atualizações periódicas do modelo, o que geralmente requer semanas ou até meses para ser concluído. Como resultado, as abordagens de classificação projetados devem ser capazes de manter sua confiabilidade por períodos mais longos, enquanto um modelo atualizado ainda está sendo projetado.

O modelo proposto visa resolver esse problema implementando a tarefa de classificação com uma opção de rejeição. O modelo proposto usa o rejeitador para identificar dois tipos de predições incorretas: a ambiguidade e novidade. Por um lado, a ambiguidade se refere a decisões que o preditor não pode realizar com alta confiabilidade,

por exemplo, porque estão mais próximas do limite de decisão do preditor. Por outro lado, a novidade trata de decisões que são muito diferentes daquelas experimentadas durante o treinamento do modelo, por exemplo, porque são um novo tipo de comportamento. A Figura 8 mostra o pipeline de classificação com o módulo rejeitador adicionado. Ele segue um pipeline tradicional de detecção de intrusão em redes de computadores com um módulo rejeitador adicionado implementado antes do módulo de alerta.

Um modelo de aprendizado de máquina com uma opção de rejeição combina o preditor utilizado w com um rejeitador r para implementar a função de decisão de rejeição em um evento de entrada dado. Dado um modelo w que produz um par de valores de confiança de classificação $\boldsymbol{\alpha = \{\alpha normal, \alpha ataque\}}$ para cada evento de entrada $\boldsymbol{x}$. Onde α mede a confiança do evento em ser das classes normal ou ataque, respectivamente, de forma que $\boldsymbol{\alpha \in R[0,1]}$. O objetivo do rejeitador $\boldsymbol{r}$ é rejeitar ou aceitar a classificação com base nos valores de confiança associados $\boldsymbol{\alpha}$, de acordo com a seguinte função: onde $\boldsymbol{\emptyset}$ indica eventos de entrada provavelmente incorretos que o preditor realiza.

$$w(x)\begin{cases}\emptyset & if\ r(a) = rejeitador \\ w & caso\ contrário\end{cases}$$

Equação 3

Como parte do problema, é preciso identificar e abordar adequadamente as classificações incorretas feitas pelo preditor devido ao comportamento em evolução do tráfego de rede. Assim, é avaliado os valores de confiança de classificação α do preditor para implementar o rejeitador de forma agnóstica em relação ao classificador, para alcançar esse objetivo. A Equação 3 mostra a função de implementação do rejeitador em nossa proposta. onde t denota o par de valores de limite de aceitação para cada classe, de forma que $\boldsymbol{t \in R[0,1]}$, e d é uma função de decisão adaptada ao módulo rejeitador proposto.

$$r(a,t)\begin{cases}\emptyset & if\ a \leq t \\ a & caso\ contrário\end{cases}$$

Equação 4

$$d(a,t)\begin{cases}r(a_{normalr}, t_{normal}) & if\ a_{normalr} > t_{ataquer} \\ r(a_{ataque}, t_{ataque}) & caso\ contrário\end{cases}$$

Equação 5

Assim, a aceitação dos eventos de entrada é estabelecida de acordo com seu nível de confiança de classificação $\boldsymbol{\alpha}$ avaliado em relação a um limite de aceitação t. Classificações com baixa confiança têm maior probabilidade de estar relacionadas a eventos de ambiguidade ou novidade, que são assumidos como ocorrendo devido ao comportamento em evolução do tráfego de rede. Portanto, classificações com baixa confiança são rejeitadas pelo sistema e adequadamente armazenadas para atualizações futuras do modelo. Em contraste, classificações com alta confiança são assumidas como eventos de rede conhecidos pelo modelo de ML implantado, portanto, podem ser aceitos com confiabilidade pelo sistema. Portanto, Algoritmo 2 - Pipeline de Classificação Confiável.

A proposta resolve a seguinte equação para implementar a otimização do limite de aceitação do rejeitador, onde ***T(w, D, y)*** é uma função de busca de limite de aceitação para um modelo de aprendizado de Máquina w, em um conjunto de dados ***D***, com um rótulo associado ***y***. O objetivo da função é encontrar um limite de aceitação ***t*** que minimize a soma das funções ***E(w, D, y, t)*** e ***R(w, D, y, t)***, onde ***E*** mede a taxa de erro nos eventos aceitos usando o limite ***t***, e ***R*** mede a taxa de rejeição com o limite utilizado. As taxas de erro e rejeição são multiplicadas por valores $\boldsymbol{\beta}$ e $\boldsymbol{\gamma}$ previamente definidos com base nas necessidades do operador de rede. Um limite de rejeição mais alto pode melhorar a confiabilidade do sistema, mas aumenta o número de eventos rejeitados como contrapartida. Por outro lado, um limite de rejeição mais baixo diminui o número de eventos rejeitados, mas afeta a confiabilidade do sistema ao longo do tempo. Nossa ideia é identificar eventos de rede desconhecidos pelo modelo de ML implantado e atualizá-lo com base no comportamento recém-identificado. Como resultado, pode-se reduzir significativamente os custos computacionais durante as atualizações do modelo e reduzir o número de eventos de rede que requerem rotulagem.

$$T(w, D, y) = \underset{t^{x2} \epsilon \mathbb{R}[0,1]}{arg\ min}\ \beta E(w, D, y, t) + \gamma R(w, D, y, t)$$

Equação 6

O pipeline de classificação proposto emprega uma implementação tradicional de sistemas de detecção de intrusão baseados em aprendizado de máquina nos peers de FL (Figura 8, Classificação Confiável). Nesse contexto, os eventos de rede são coletados por

meio de um módulo de Aquisição de Dados, enquanto o comportamento dos eventos coletados é extraído por um módulo de Extração de Características. Um módulo de Classificação usa o vetor de características construído como entrada, que aplica o modelo de ML e gera um valor de confiança de classificação associado. O valor de confiança de classificação é utilizado como entrada pelo módulo Rejeitador, que avalia se um dado limite de confiança de classificação é atendido de acordo com a Equação 3. Classificações com alta confiança são aceitas pelo modelo proposto e podem ser usadas para acionar alertas. Por outro lado, classificações com baixa confiança são rejeitadas pelo modelo proposto e armazenadas em um conjunto de dados $D^{\emptyset}_{i}$ para futuras atualizações do modelo dentro de cada peer de FL.

Require:
Model w_G
Event $x = \{f_1, f_2, ..., f_N\}$
Threshold $t = \{t_{attack}, t_{normal}\}$
procedure CLASSIFICATION(w_G, x, t)
$\alpha \leftarrow classify(w_G, x)$
if $\alpha_{normal} > \alpha_{attack}$ **and** $\alpha_{normal} \geq t_{normal}$ **then**
$alert(x, normal)$
else if $\alpha_{attack} \geq t_{attack}$ **then**
$alert(x, attack)$
else
$reject(x)$
end if
end procedure

Algoritmo 2 - Pipeline de Classificação Confiável

O Algoritmo 2 mostra o processo de pipeline de classificação confiável (Figura 8, Classificação Confiável). Ele recebe como entrada um modelo de aprendizado de máquina ***w***, uma instância ***x*** representada por um conjunto de características ***f*** e um limite de aceitação ***t*** associado a cada classe *(normal* ou *ataque*). O modelo classifica a instância de entrada, gerando um par de valores de confiança de classificação ***α***. De acordo com a confiança de classificação, o evento aciona alertas se ultrapassar um limite de rejeição previamente definido. Caso contrário, o evento é rejeitado e usado no pipeline de atualização do modelo.

Como resultado, o pipeline de classificação proposto pode garantir a confiabilidade do sistema ao longo do tempo, mesmo na presença de um modelo de ML desatualizado. No entanto, a abordagem proposta também pode identificar proativamente novos tipos de comportamento de tráfego de rede que devem ser utilizados para atualizações do modelo. Isso ocorre porque modelo rejeitará novos dados de tráfego de rede de forma não supervisionada devido à baixa confiança de classificação.

5.2.2. Atualização do Modelo baseado em FL

A atualização do modelo é uma tarefa desafiadora em sistemas de detecção de intrusões baseados em aprendizado de máquina (*NIDS – Network Intrusion detection System*). Isso se deve ao fato de que as atualizações do modelo exigem o fornecimento de um conjunto de dados rotulados atualizado e a execução de um processo de treinamento de modelo computacionalmente caro. Por um lado, construir um novo conjunto de dados de treinamento geralmente requer a coleta de novos comportamentos de tráfego de rede e assistência humana para rotulação adequada. Por outro lado, o treinamento do modelo é um processo geralmente computacionalmente caro.

A abordagem proposta implementa atualizações periódicas do modelo em um processo de três fases para enfrentar esse desafio. Primeiro, as atualizações do modelo são conduzidas por meio de uma lógica de FL, melhorando a qualidade do modelo construído enquanto permite que os operadores de rede aproveitem as CTI das organizações participantes. Em segundo lugar, as atualizações do modelo são realizadas usando os eventos previamente rejeitados pelo sistema. Consequentemente, podemos reduzir significativamente o número de eventos que devem ser usados para as atualizações do modelo. A principal suposição é que as atualizações do modelo podem ser realizadas apenas usando os eventos que o modelo de ML não conseguiu classificar de forma confiável, ajustando o sistema à medida que o tempo passa. Em terceiro lugar, as atualizações do modelo são realizadas seguindo uma abordagem de aprendizado por transferência, usando o modelo desatualizado para reduzir os custos computacionais e aproveitar o conhecimento previamente treinado disponível no modelo.

O pipeline proposto para atualização do modelo baseado em FL é mostrado na Figura 8. Ele considera um servidor central "***s***" e um conjunto de "***k***" Peers P_i. Assume-se que o servidor central é benigno e é responsável por conduzir as tarefas de Inicialização, Distribuição e Agregação. Cada peer P_i é responsável por conduzir a tarefa de Treinamento Local e possui um conjunto de dados não confiáveis $D^{\emptyset}_i$ que armazena os eventos anteriormente rejeitados pelos pares desde a última execução da atualização do modelo. Cada peer P_i é responsável por conduzir a rotulação do Conjunto de Dados Confiáveis para compor um Conjunto de Dados Rotulado D_i. A rotulação do conjunto de dados pode ser feita por ferramentas automatizadas de detecção de intrusões baseadas em uso indevido ou por meio da assistência do operador de rede.

A tarefa de atualização do modelo é executada periodicamente, por exemplo, a cada mês, pelo servidor central "***s***" (Figura 8, Acionador de Atualização Global). O processo inicialmente distribui o modelo global da rodada w^t_G para todos os peers do FL. Sem compartilhar seus dados privados, cada peer P_i atualiza seu modelo local de acordo com seus eventos anteriormente rejeitados e rotulados D_i. O conjunto de dados rotulado D_i é usado para atualizar o modelo W^t_G e compor um modelo local W^t_i seguindo uma lógica de *transfer learning*, ajustando o modelo aos eventos que foram anteriormente rejeitados pelo pipeline de classificação confiável do peer. O modelo local atualizado é retornado ao servidor central, que realiza a tarefa de agregação do modelo, resultando em um modelo global. O processo de atualização do modelo é repetido até que um determinado número de rodadas de execução T seja alcançado. Por fim, o modelo global construído W_G é enviado de volta para todos os peers do FL, finalizando a tarefa de atualização do modelo.

Require:
Model w_G^t ▷ Global Round Model
Unreliable Dataset $\mathcal{D}_i^{\emptyset}$ ▷ Peer i rejected events
Label Provider l ▷ Network Operator
procedure PEERUPDATE(w_G^t, $\mathcal{D}_i^{\emptyset}$, l)
$\mathcal{D}_i \leftarrow label(\mathcal{D}_i^{\emptyset}, l)$ ▷ Label rejected dataset
$w_i^t \leftarrow update(w_G^t, \mathcal{D}_i)$ ▷ Update model
$export(w_i^t)$ ▷ Export to central server
end procedure

Algoritmo 3 - Processo de Atualização do Modelo

O Algoritmo 3 mostra o processo de atualização do modelo. Ele é executado em cada peer participante, em cada execução de atualização do modelo da rodada global, recebendo como entrada um modelo W^t_G, um conjunto de dados não confiáveis $D^{\emptyset}_i$ contendo os eventos rejeitados pelo peer ***i*** e um provedor de rótulos "***l***". Vale lembrar que o provedor de rótulos pode ser realizado usando ferramentas automatizadas de detecção de intrusões baseadas em uso indevido ou com a assistência do operador de rede. O provedor de rótulos "***l***" rotula o conjunto de dados rejeitado $D^{\emptyset}_i$ para compor um conjunto de dados rotulado D_i. O conjunto de dados rotulado D_i atualiza o modelo recebido, resultando em um modelo local W^t_i. Finalmente, o classificador atualizado pelo peer W^t_i é enviado para o servidor central para agregar um modelo global correspondente.

Como resultado, os benefícios da tarefa de atualização do modelo baseada em FL são duplos. Primeiro, os participantes não precisam compartilhar seus dados privados, ao mesmo tempo em que podem utilizar o conhecimento de outros participantes, representado por seus modelos locais. Portanto, vários participantes podem colaborar durante a tarefa de atualização do modelo sem envolver uma grande quantidade de dados de treinamento. Segundo o processo de atualização do modelo é baseado nos eventos rejeitados ao longo do tempo. Assim, os dados de treinamento usados nas atualizações diminuem significativamente, enquanto o modelo é ajustado de acordo com os eventos anteriormente rejeitados. Essa característica permite que os operadores ajustem seu modelo de ML implantado com base no tráfego de rede recente.

5.2.3. Discussão

A proposta, traz diversos assuntos em relação a sistemas de detecção de intrusão baseados em aprendizado de máquina e a necessidade de lidar com a evolução constante do comportamento do tráfego de rede no mundo real. A instabilidade e a variabilidade desse comportamento tornam desafiadora a manutenção da confiabilidade dos modelos de detecção de intrusão ao longo do tempo. Em resposta a esse desafio, a proposta apresentada neste estudo introduz um inovador pipeline de classificação confiável, que incorpora um módulo de rejeição capaz de identificar eventos de baixa confiança, relacionados à ambiguidade ou novidade, em um ambiente de aprendizado federado. Esse módulo permite a atualização proativa do modelo com novos comportamentos de rede,

reduzindo o tempo e os recursos necessários para as atualizações, ao mesmo tempo em que mantém a confiabilidade do sistema.

O pipeline de classificação confiável é uma abordagem promissora para lidar com a mudança constante do tráfego de rede em ambientes de detecção de intrusão baseados em aprendizado de máquina. Essa abordagem não apenas se preocupa com a confiabilidade do sistema em mudanças no comportamento da rede, mas também contribui para a eficiência operacional, minimizando o tempo e os custos associados às atualizações do modelo. Portanto, a pesquisa aqui apresentada oferece uma perspectiva promissora para aprimorar a detecção de intrusões em redes em ambientes dinâmicos, abrindo caminho para avanços significativos na área de segurança cibernética.

Em conclusão, a atualização contínua dos modelos em sistemas de detecção de intrusões baseados em aprendizado de máquina (ML-NIDS) é um desafio complexo, devido à necessidade de conjuntos de dados rotulados atualizados e ao alto custo computacional envolvido no treinamento dos modelos. A abordagem proposta neste estudo oferece uma solução inovadora para esse problema, implementando atualizações do modelo de forma eficiente e econômica. Enquanto a estratégia de atualização do modelo baseada em FL permite que os operadores de rede aprimorem a qualidade de seus modelos sem comprometer a privacidade dos dados. Ao distribuir o modelo global para os peers participantes, cada um pode atualizar seu modelo local usando os eventos anteriormente rejeitados, reduzindo significativamente a quantidade de dados necessária para a atualização. Além disso, a tarefa de rotulagem dos dados é facilitada com a assistência de ferramentas automatizadas de detecção de intrusões ou operadores de rede.

Capítulo 6

Avaliação

A avaliação da proposta tem como objetivo responder às seguintes perguntas de pesquisa:

- (RQ3) Como o modelo de aprendizado por reforço proposto se comporta sem atualizações de modelo?
- (RQ4) Qual é o impacto das atualizações de modelo na proposta de aprendizado por reforço?
- (RQ5) O modelo de aprendizado por reforço proposto pode fornecer maior confiabilidade do que as técnicas tradicionais?
- (RQ6) Como o modelo de aprendizado federado tradicional se comporta ao lidar com novos dados de tráfego de rede?
- (RQ7) Como a técnica de rejeição proposta no aprendizado federado auxilia na melhoria da confiabilidade da classificação?
- (RQ8) Como o modelo de aprendizado federado proposto se comporta ao longo do tempo com as atualizações periódicas do modelo?

As subseções a seguir descrevem detalhadamente o procedimento de construção dos modelos propostos e sua avaliação.

6.1. Avaliação Aprendizado por Reforço

As subseções a seguir descrevem detalhadamente o procedimento de construção do modelo proposto e avaliação do Aprendizado por Reforço, além de responder as questões de pesquisa RQ3, RQ4 e RQ5.

6.1.1. Construção do Modelo

A proposta de aprendizado por reforço para detecção de intrusões (Algoritmo 1) foi implementada em cima da API OpenAI Gym [48]. A cada treinamento ou atualização do modelo, o algoritmo cria um ambiente de teste que reproduz o procedimento de treinamento proposto (ver Seção 5.1). A proposta utiliza um *Multi-Layer Perceptron* (MLP) como política de aprendizado por reforço (ou seja, modelo de ML).

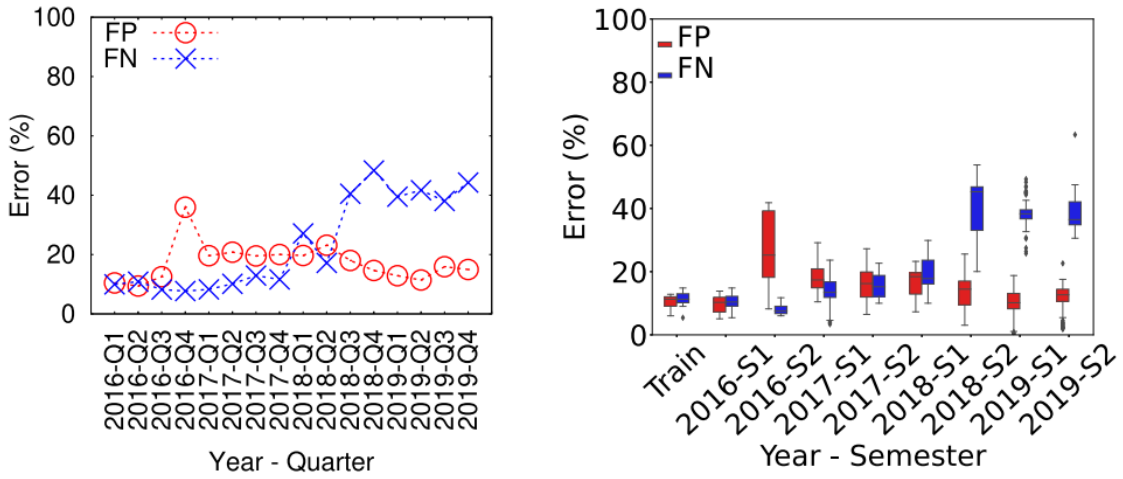
O algoritmo permite que o procedimento de atualização do modelo proposto (ver Seção 5.1) seja executado sobre os neurônios do MLP antigo através de um procedimento de *transfer learning*. Os dados de janeiro de 2016 foram usados (30 dias) para o primeiro treinamento do modelo, permitindo a subsequente avaliação das atualizações do modelo, que se basearão no agente desatualizado. O procedimento de atualização do modelo utiliza apenas dados de 7 dias (Figura 7, Banco de Dados com Janela Deslizante de Eventos). Assim, foi realizado atualizações do modelo usando menos dados do que os avaliados anteriormente para avaliar adequadamente como modelo se comporta com menos dados de treinamento durante as atualizações.

O MLP com algoritmo proposto é executado em cada atualização do modelo com uma taxa de aprendizado de 0,3, uma taxa de momento para o algoritmo de retropropagação de 0,2, e usa a API TensorFlow [49]. O algoritmo executa 5.000 épocas para construir o modelo, onde cada época realiza 100 turnos. Cada turno calcula os gradientes de política do algoritmo proposto de acordo com as recompensas obtidas (ver Eq. (2)) a partir da classificação de 10 mil instâncias de treinamento.

Como critério de convergência durante a atualização do modelo, o algoritmo executa 5.000 épocas ou atinge 90% de precisão. Esses parâmetros foram identificados empiricamente, resultando em resultados de classificação semelhantes.

6.1.2. Acurácia da Classificação Sem Atualizações

O primeiro experimento visa responder à RQ3 e avalia o desempenho do modelo proposto quando nenhuma atualização do modelo é realizada ao longo do tempo. Semelhante ao que foi feito anteriormente, foi aplicado o algoritmo proposto para criar uma política de agente usando dados de janeiro de 2016 do MAWIFlow como o ambiente de treinamento (Figura 7). O agente obtido é usado em todos os dados do MAWIFlow sem atualizações do modelo. A Figura 9-A mostra o desempenho médio de erro considerando as taxas de FP e FN do modelo proposto em uma periodicidade de três meses.



A) Acurácia do modelo ao longo do tempo B) Distribuição das diária das acurácias

Figura 9 - Desempenho d acurácia da proposta ao longo do tempo, executado no conjunto de dados MAWIFlow. A proposta é treinada com dados de janeiro de 2016 e não é atualizada ao longo do tempo.

O modelo proposto manteve sua confiabilidade por longos períodos, mantendo as taxas de FP próximas às medidas na fase de treinamento do classificador ao longo dos quatro anos. Por exemplo, nossa técnica proposta apresentou uma taxa média de erro de 18,9% e 8,8% para FP e FN, considerando o primeiro ano de implantação (ou seja, 2016). Em nossa proposta, cada mês após o período de treinamento, em média, aumenta 4,2% e 0,3% nas taxas de FP e FN, respectivamente, considerando uma duração do modelo de 1 ano. Além disso, o modelo proposto apresentou taxas médias de FP e FN ao longo dos quatro anos de 16,4% e 23,5%.

O modelo alcançou taxas de precisão semelhantes às obtidas por técnicas tradicionais, como o classificador RF com uma duração de 6 meses (Figura 4), que apresentou taxas de 16,5% e 23,4% de FP e FN. Portanto, o modelo forneceu alta precisão de detecção mesmo quando não ocorreram atualizações do modelo, com taxas de precisão semelhantes às obtidas pelas técnicas da literatura com uma duração do modelo de 6 meses.

Também investigamos como a precisão do modelo proposto varia ao longo do tempo sem atualizações do modelo, conforme mostrado na Figura 9-B A variação na taxa de precisão do modelo é significativamente menor quando comparada às abordagens tradicionais de ML. O método proposto apresentou intervalos interquartis médios em 2016-S1 de apenas 3,4% e 1,9% de FP e FN, respectivamente. Em contraste, abordagens tradicionais (por exemplo, RF) apresentaram um intervalo interquartil médio em 2016-S1 de 3,1% e 6,1% nas taxas de FP e FN, respectivamente. Além disso, se considerarmos todo o tráfego de 4 anos do MAWIFlow, nossa proposta apresenta um intervalo interquartil médio de 13,5% e 13,3% nas taxas de FP e FN, enquanto o classificador RF apresenta 29,3% e 36,6% de intervalo interquartil (Figura 3), respectivamente em ambos os casos. Consequentemente, nossa abordagem proposta aumenta a duração dos modelo de detecção de intrusão ao melhorar a precisão dos sistemas e reduzir a variação da precisão ao longo do tempo.

6.1.3. Acurácia da Classificação com Atualizações

Para responder à RQ4, realizamos atualizações periódicas do modelo proposto. Nesse caso, similarmente aos experimentos realizados na Seção 4, executa o procedimento de atualização proposto a cada semestre. No entanto, levando em conta apenas os eventos ocorridos nos últimos sete dias (Figura 7, Banco de Dados com Janela Deslizante de Eventos). Como o modelo aproveita o conhecimento anterior através do modelo desatualizado em um estilo de aprendizado por transferência (Seção 5.1), avaliamos inicialmente como o modelo de ML desatualizado pode facilitar o procedimento de atualização do modelo.

A Figura 10 mostra a convergência da proposta de acordo com a época de atualização do modelo, considerando se o modelo desatualizado é utilizado ou não no segundo semestre de 2016. A abordagem proposta que aproveita o modelo desatualizado converge demandando significativamente menos épocas para ser executada, alcançando

uma taxa de precisão de 90% com apenas 970 épocas, mesmo com apenas 1 semana de dados de treinamento. Em contraste, se o modelo desatualizado não for usado, é necessário executar 4610 épocas. Resultados semelhantes foram encontrados independentemente do período em que a atualização do modelo ocorreu.

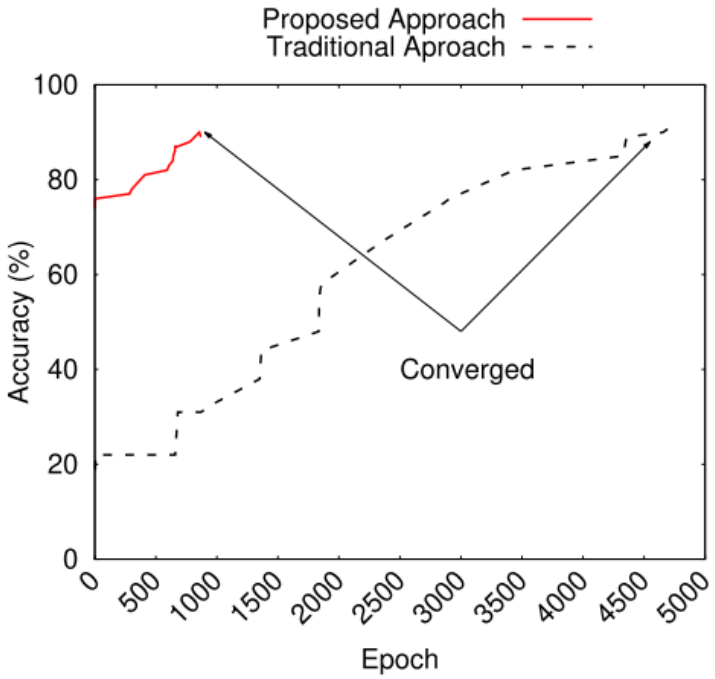
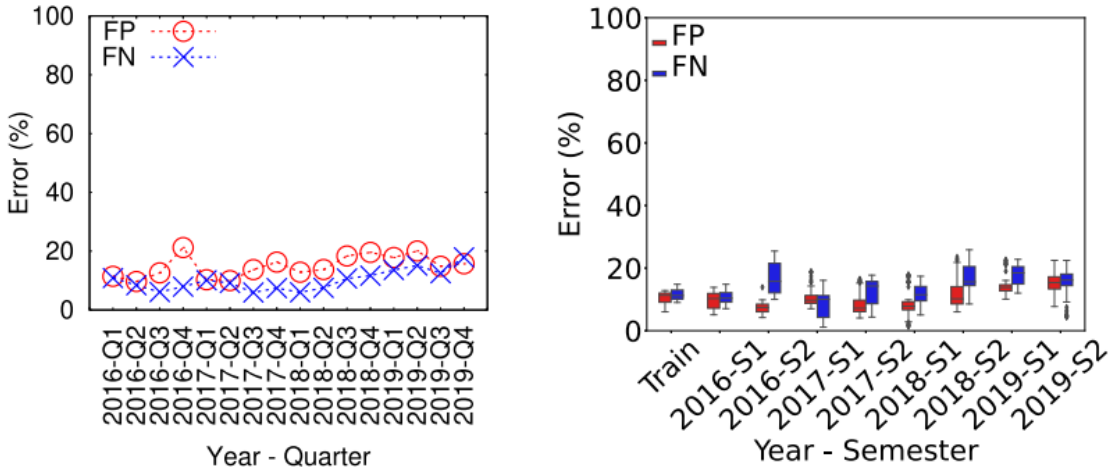


Figura 10 - Convergência do treinamento da proposta no 2º semestre de 2016, considerando um agente desatualizado usado no processo de atualização do modelo e sua comparação com o retreinamento do zero. A acurácia foi medida como a proporção de eventos totais classificados corretamente. Resultados semelhantes foram encontrados em cada execução do procedimento de atualização do modelo.

A abordagem proposta que aproveita o modelo desatualizado pode diminuir significativamente as necessidades computacionais no procedimento de treinamento devido à redução no número de épocas. Também reduz o número de eventos que devem ser rotulados devido a uma maior durabilidade do modelo, exigindo, portanto, menos atualizações do modelo. Em média, a abordagem proposta que aproveita o modelo desatualizado convergiu com apenas 21,1% das épocas, confiando em apenas uma semana de dados de treinamento em comparação com a abordagem tradicional que realiza atualizações do modelo do zero.

A Figura 11-A mostra a precisão da abordagem proposta ao longo do tempo com uma periodicidade de atualização do modelo de 6 meses. Nesse caso, as taxas de erro são significativamente mais baixas e mais estáveis ao longo do tempo. A periodicidade das atualizações do modelo permitiu que nossa proposta alcançasse altas taxas de precisão ao longo dos quatro anos de dados do MAWIFlow. A técnica proposta apresentou uma média de 14,5% e 10,0% de taxas de FP e FN ao longo do tempo, respectivamente. A Figura 11B mostra a variação da precisão ao longo do tempo com atualizações periódicas do modelo.



A) Acurácia do modelo ao longo do tempo B) Distribuição das acurácias diárias

Figura 11 - Desempenho da acurácia da proposta ao longo do tempo, executado em todo o conjunto de dados MAWIFlow. A proposta é treinada com dados de janeiro de 2016 e atualizada a cada semestre com dados de 1 semana.

A variação da precisão diminui significativamente, apresentando uma média de intervalo interquartil de apenas 4,4% e 8,5% nas taxas de FP e FN, enquanto as técnicas tradicionais de ML (por exemplo, RF) (Figura 5) apresentam uma média de 16,9% e 5,3% (ou seja, 3,84× mais FP para apenas uma redução de 0,37× em FN), respectivamente em todos os casos. Portanto, o modelo proposto aumenta significativamente a confiabilidade do modelo de detecção de intrusões ao longo do tempo, estendendo a durabilidade do modelo e reduzindo a variação da precisão enquanto mantém a precisão ao longo do tempo.

Foi Investigado ainda como a periodicidade da atualização do modelo impacta a precisão da nossa proposta. A Figura 12 mostra a relação entre a periodicidade da atualização do modelo e as taxas de precisão da proposta. Nossa proposta alcança taxas de precisão significativamente mais altas, mesmo quando a durabilidade do modelo é considerada. Mais especificamente, mesmo com uma durabilidade de modelo de 2 anos,

alcança melhor precisão de classificação do que técnicas tradicionais que consideram uma durabilidade de modelo de 1 mês (Figura 6).

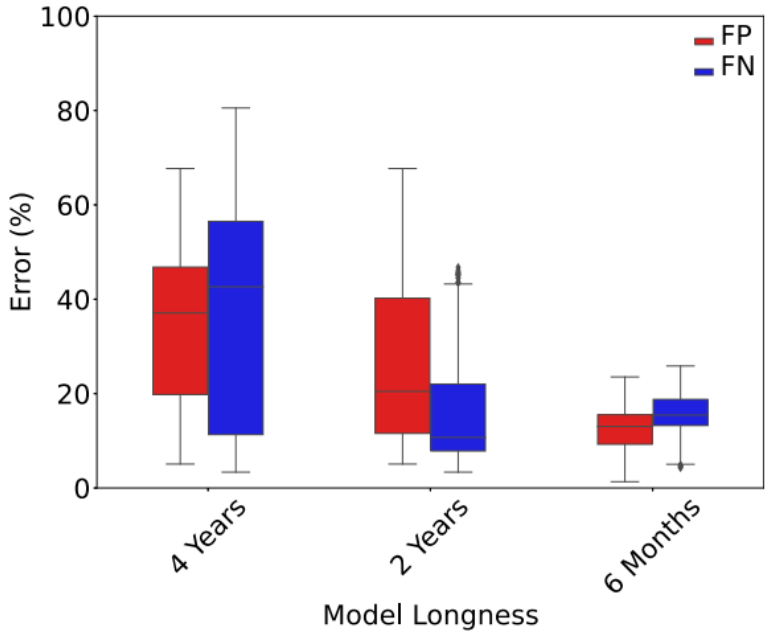
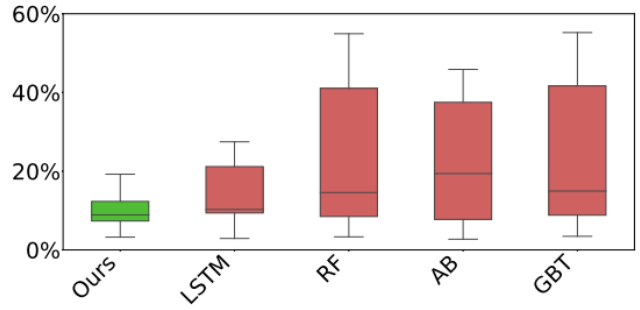


Figura 12 - A relação entre durabilidade e taxa média de erro no conjunto de dados proposto MAWIFlow. A durabilidade do modelo estabelece sua frequência de atualização, enquanto a taxa média de erro é medida como a média das taxas de FP e FN em todo o conjunto de dados MAWIFlow.

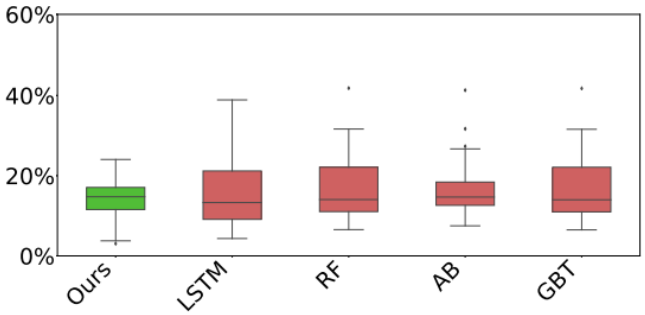
Em resumo, nossa proposta aumenta a durabilidade do modelo e diminui a periodicidade das atualizações do modelo sem degradar a precisão. Além disso, alcança taxas de precisão mais altas do que as técnicas tradicionais, mesmo quando não ocorrem atualizações do modelo.

Finalmente, para responder à questão RQ5, foi comparado as precisões obtidas pelo modelo e aquelas das técnicas tradicionais de ML. A Figura 13 mostra a distribuição mensal das taxas de precisão de cada modelo de classificação avaliado com uma durabilidade de modelo de 6 meses. Nesse caso, a mediana do modelo proposto foi de 14,3%, 8,4% e 0,92 de FP, FN e F-Measure, respectivamente. Em contraste, as técnicas tradicionais apresentaram uma mediana de F-Measure de 0,83, 0,81, 0,78 e 0,81 para LSTM, RF, Ada e GBT, respectivamente. Assim, o modelo proposto também poderia fornecer maiores precisões de detecção do que modelos de aprendizado profundo, como o LSTM avaliado, melhorando o F-Measure em 0,09 enquanto demanda apenas 20% dos

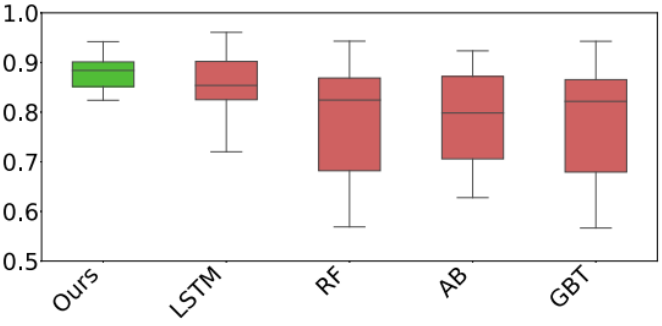
dados de treinamento necessários e significativamente menos recursos computacionais para atualizações do modelo.



A) Falso negativo



B) Falso positivo

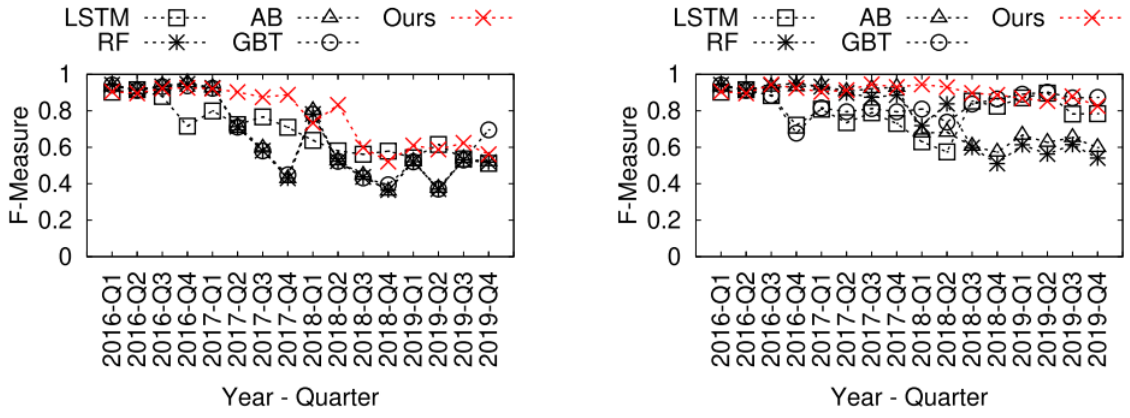


C) F-Measure

Figura 13 - Acurácias mensais para várias técnicas de classificação considerando uma periodicidade de atualização do modelo de 6 meses, executado em quatro anos de dados do MAWIFlow. O modelo proposto pode fornecer intervalos interquartis menores e alcançar uma acurácia mediana maior do que outras técnicas avaliadas.

A Figura 14 mostra uma comparação de desempenho do modelo proposto com as abordagens previamente avaliadas. É possível notar que a proposta pode melhorar significativamente a durabilidade do modelo (ver Figura 14-A). Por exemplo, a proposta proporciona uma média de F-Measure ao longo de 2016 e 2017 de 0,91 sem atualizações periódicas do modelo, enquanto o classificador RF alcança apenas 0,78 (ou seja, uma melhoria de 0,13). Além disso, se atualizações periódicas do modelo forem realizadas, o modelo proposto alcança uma média de F-Measure de 0,92 ao longo dos 4 anos do

MAWIFlow, enquanto o classificador RF, como exemplo, proporciona uma média de F-Measure de apenas 0,80. Como resultado, o modelo proposto pode fornecer maiores precisões de classificação na maioria das vezes (meses), melhorando a taxa média de FP em até 8,0% e a taxa média de FN em até 34,6% quando comparado a técnicas tradicionais.



A) Sem atualizações

B) Com atualizações

Figura 14 - Comparação do desempenho da acurácia da proposta ao longo do tempo com e sem atualizações periódicas do modelo, executada em todo o conjunto de dados MAWIFlow.

6.1.4. Limitações e Desafios

A detecção de intrusões baseada em rede através de técnicas de ML na literatura ignora os desafios relacionados ao comportamento evolutivo do tráfego de rede, focando seus esforços de pesquisa em obter maiores precisões, o que muitas vezes é alcançado apenas com um comprometimento na vida útil do modelo. O modelo proposto melhorou significativamente a vida útil do modelo de ML, proporcionando maior precisão após o período de treinamento do que as técnicas tradicionais. No entanto, apesar da melhoria significativa na precisão média, as taxas de FP e FN obtidas, devido à enorme quantidade de tráfego de rede, ainda podem representar um desafio significativo para a implantação do modelo proposto em ambientes de produção. Várias técnicas podem ser usadas para diminuir as taxas de FP ou FN. Por exemplo, o administrador pode mudar o ponto de operação do classificador para aumentar a periodicidade de atualização do modelo. Ela pode usar um conjunto de classificadores e correlacionar os alarmes disparados de acordo

com a fonte do host, pois cada fluxo de rede pode disparar um alarme correspondente. Assim, um único ataque pode gerar milhares de alarmes, por exemplo, negação de serviço baseada em rede.

6.1.5. Considerações Finais

Abordagens inovadoras para a detecção de intrusões através de técnicas de aprendizado de máquina foram amplamente propostas na literatura científica recentemente, enquanto apenas algumas foram implantadas em produção. Mostramos no artigo que os pesquisadores adotam incorretamente suposições tradicionais de aprendizado de máquina no domínio da detecção de intrusões, como assumir um comportamento estático do tráfego de rede.

Este artigo propôs a evolução dessas suposições de detecção de intrusões em relação ao comportamento do tráfego de rede. Até onde sabemos, a proposta é a primeira a construir um conjunto de dados que avalia a confiabilidade dos modelos de detecção de intrusões ao longo do tempo. Além disso, propusemos uma nova técnica de aprendizado por reforço para a detecção de intrusões com maior vida útil do modelo, menor variação de precisão e atualizações de modelo mais acessíveis. A técnica proposta proporcionou maior precisão de detecção, mesmo quando não ocorrem atualizações do modelo, e, quando ocorrem, a proposta diminui significativamente a variação de precisão enquanto também melhora a precisão da detecção.

Como trabalhos futuros, os pesquisadores são encorajados a usar o conjunto de dados construído para aumentar a precisão do sistema enquanto proporcionam uma maior vida útil do modelo. Portanto, as pesquisas conduzidas no conjunto de dados devem considerar o equilíbrio entre precisão do modelo, vida útil do modelo, periodicidade de atualização do modelo e número de amostras de treinamento. O conjunto de dados construído e utilizado nos experimentos deste artigo está disponível publicamente para download em https://secplab.ppgia.pucpr.br/reinforcemawiflow.

6.2. Avaliação Aprendizado Federado

As subseções a seguir descrevem detalhadamente o procedimento de construção do modelo proposto e avaliação do Aprendizado Federado, além de responder as questões de pesquisa RQ5, RQ6 e RQ7.

6.2.1. Construção do Modelo

O modelo proposto é avaliado utilizando um classificador Perceptron Multicamadas (MLP) implementado com 500 neurônios ocultos, uma função de ativação relu e otimização adam. Os parâmetros são definidos empiricamente. O classificador é implementado através da API scikit-learn v0.24 [75]. Adotamos o MLP para a implementação de FL, agregando os modelos locais em um único modelo global. Assim, implementamos a agregação do modelo global computando a média dos pesos locais do MLP por meio do algoritmo FedAVG.

A seguir, é descrito as etapas do processo de treinamento e atualização de modelo implementado baseado em FL (Figura 8):

1) Inicialização. Se for uma inicialização proposta, o servidor central s inicializa aleatoriamente um modelo global. Caso contrário, se for realizada uma atualização do modelo, o modelo global treinado anteriormente é utilizado.

2) Distribuição. O modelo global é enviado para todos os pares de FL.

3) Treinamento Local. Cada par executa o Algoritmo 3 e atualiza o modelo recebido para compor um modelo. O modelo é ajustado usando 500 épocas. Se for o treinamento inicial, cada par atualiza o modelo recebido com eventos selecionados aleatoriamente do conjunto de dados de treinamento. Caso contrário, se for realizada uma atualização do modelo, o treinamento é realizado com base nos eventos rejeitados do par armazenados no conjunto de dados.

4) Agregação. O modelo construído é enviado para o servidor central. O servidor central executa o algoritmo FedAVG para compor um modelo global. Outra rodada do treinamento ou atualização do modelo é executada (fases 2 a 4). Caso contrário, o

treinamento do modelo é finalizado, e o modelo global construído é enviado para os pares para fins de detecção de intrusões.

Consideramos um cenário de implantação proposta com 10 pares de FL (Fig. 8, Pares). Adotamos uma técnica de subamostragem aleatória sem reposição para equilibrar a ocorrência entre as classes durante o procedimento de treinamento inicial.

6.2.2. Aprendizado Federado Confiável

O primeiro experimento tem como objetivo responder a RQ6 e investigar o desempenho de acurácia do FL tradicional. É executado o procedimento de treinamento considerando um cenário com 10 pares, implementado através do FedAVG, sem utilizar o trabalho proposto. Executa o FL tradicional dividindo aleatoriamente, de maneira estratificada, os dados de janeiro entre os pares. Cada par constrói seu modelo local, enquanto o modelo global é construído pelo servidor central com base nos modelos locais de cada par (consulte a Seção 5.2).

Figura 15 - Desempenho de classificação da implementação FL tradicional sem atualizações do modelo ao longo do tempo. O sistema só é treinado com dados de janeiro.

A Figura 15 mostra a acurácia de classificação do modelo de FL tradicional sem atualizações periódicas do modelo. Em comparação com as técnicas tradicionais nota-se uma diminuição na precisão do modelo FL à medida que o tempo passa e não realizamos atualizações do modelo. Por exemplo, a abordagem de FL tradicional não atualizado alcança uma taxa de FP de 64% em junho Figura 15. Da mesma forma, a diminuição da precisão é experimentada um mês após a fase de treinamento, aumentando a taxa de FP

em 22%. Como resultado, o FL tradicional não consegue manter sua confiabilidade à medida que o tempo passa, exigindo a execução de atualizações periódicas do modelo.

O segundo experimento tem como objetivo responder a RQ7 e avaliar como nossa técnica de verificação proposta pode melhorar a acurácia de classificação do sistema, mesmo sem realizar o procedimento de atualização do modelo. Para alcançar esse objetivo, foi implementado a otimização do limiar do rejeitador proposto, considerando uma variação no limiar em um intervalo de 0,01. Calculando o valor de confiança do classificador MLP através da função predict_proba da API scikit-learn. Como o ponto de operação do rejeitador deve ser definido de acordo com as necessidades do operador da rede, implementamos a otimização considerando $\beta = 1.0$ e $\gamma = 1.0$. Portanto, rejeição e erro têm o mesmo peso durante o processo de busca do limiar. Primeiramente, avaliamos a troca entre erro e rejeição em fevereiro usando o modelo treinado em janeiro (mostrado na Figura 15).

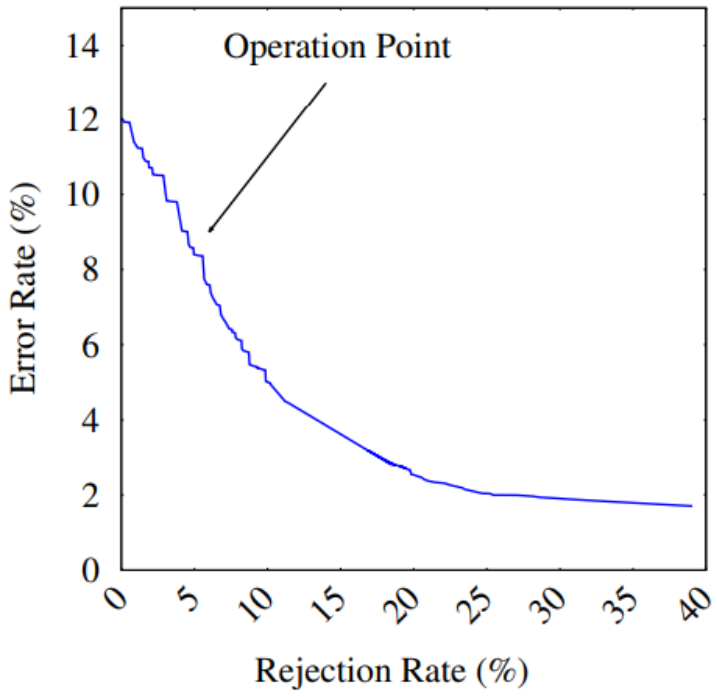


Figura 16 - Taxa de erro e rejeição da técnica de verificação de classificação proposta. O classificador é treinado nos dados de janeiro e avaliado em fevereiro.

A Figura 16 mostra a troca entre taxa de erro e taxa de rejeição em fevereiro quando foi adotado a técnica de rejeição proposta usando a abordagem de busca de limiar implementada. É possível observar que a técnica de verificação proposta pode melhorar

a precisão de detecção ao avaliar os valores de confiança de classificação. Por exemplo, o operador da rede pode reduzir a taxa de erro do sistema em 10% se puder tolerar uma taxa de rejeição de 20%. No entanto, uma rejeição de apenas 5% pode reduzir a taxa de erro do sistema em 4%. Como resultado, nossa técnica de verificação proposta pode ser usada para garantir a confiabilidade do sistema à medida que o tempo passa, mesmo quando as atualizações do modelo não são realizadas.

Aproveitando a técnica de verificação, foi avaliado a precisão de detecção dos modelos se não assumir a realização de atualizações do modelo. De fato, foi usado a técnica de verificação para realizar essa tarefa e suprimir classificações com valores de confiança baixos durante o cálculo da precisão. Para alcançar esse objetivo, consideramos um ponto de operação de rejeição de 5%, que utiliza um αnormal de 0,92 e αataque de 0,87. É importante observar que o ponto de operação de rejeição deve ser definido de acordo com as necessidades do operador da rede. Por um lado, pode-se melhorar a precisão de detecção rejeitando mais instâncias. Por outro lado, pode-se classificar instâncias adicionais se tolerar um certo grau de taxa de erro. A avaliação considera um cenário realista em que o operador da rede tolera apenas 5% de eventos rejeitados.

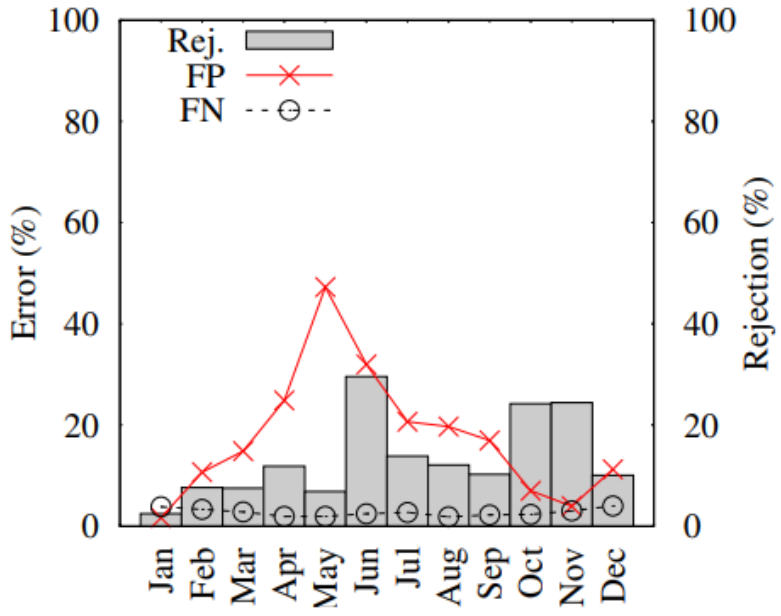


Figura 17 - Modelo proposto sem atualizações por meio de verificação.

A Figura 17 mostra a precisão de classificação da proposta sem atualizações periódicas do modelo, utilizando o módulo do rejeitador com os limiares de aceitação previamente selecionados. O modelo proposto pode melhorar a precisão de detecção ao avaliar a confiabilidade da classificação do evento de entrada. Em detalhes, a técnica de verificação pode aumentar a taxa média de FP em 12,9%, rejeitando apenas uma média

de 13,5% das instâncias quando comparado com seu equivalente sem atualização. O módulo verificador proposto pode manter o sistema confiável por intervalos estendidos, mesmo quando as atualizações do modelo não são consideradas.

Para responder à RQ8, foi investigado ainda mais como a execução de atualizações periódicas do modelo com base em eventos rejeitados pode melhorar a precisão do sistema. Para esse fim, foi realizado atualizações mensais do modelo baseado em FL de acordo com os eventos rejeitados por cada par (Figura 8, conjunto de dados não confiáveis (Di∅)). Ao mesmo tempo, o servidor central s realiza a tarefa de agregação conforme retratado pelo algoritmo FedAVG. Da mesma forma, as atualizações do modelo são executadas mensalmente, conforme avaliado nas abordagens tradicionais.

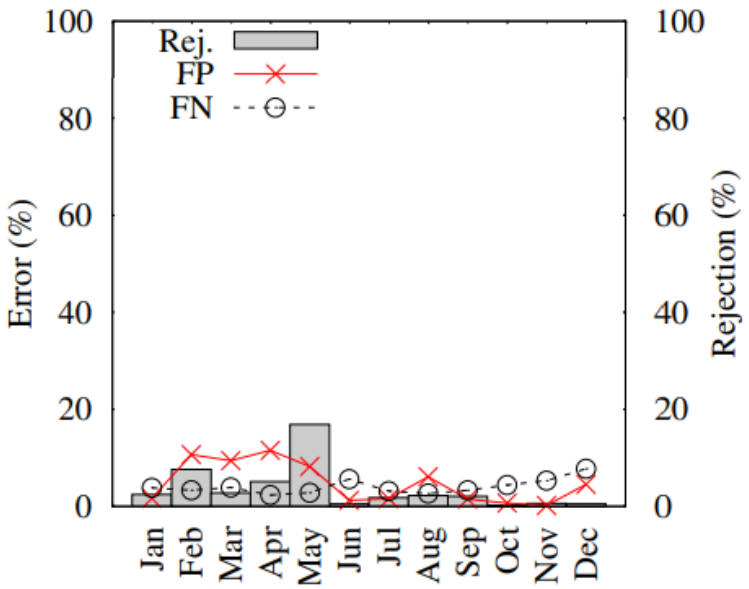
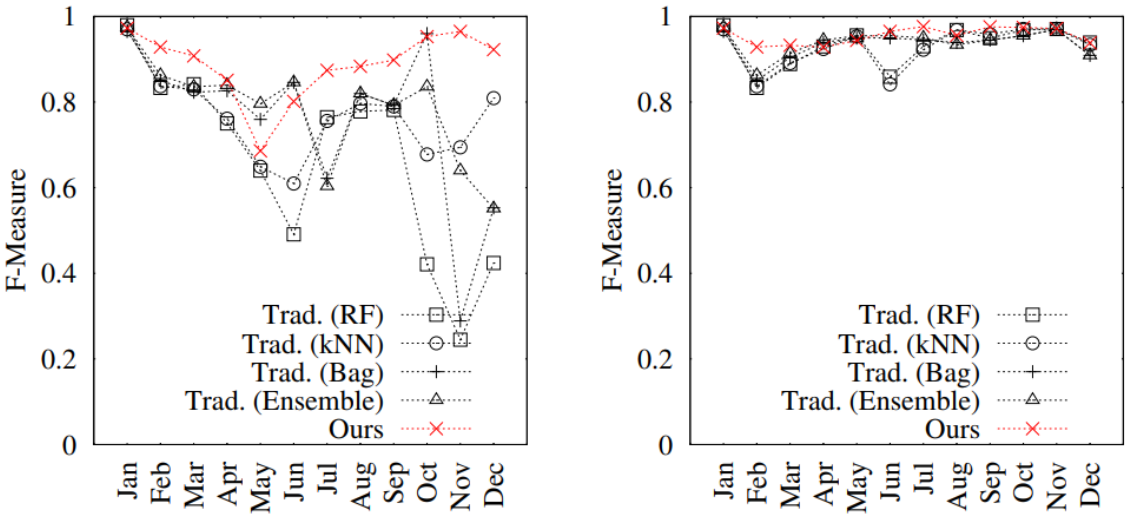


Figura 18 - Modelo proposto com atualizações e verificação mensais.

A Figura 18 mostra a precisão de classificação do modelo com atualizações mensais periódicas do modelo. A abordagem proposta melhora significativamente a precisão de detecção enquanto reduz o número de eventos rejeitados. Mais especificamente, em comparação com seu equivalente sem atualização (Figura 17 vs. Figura 18), a abordagem proposta pode fornecer uma taxa média de FP de apenas 3,3%, rejeitando apenas 3,6% das instâncias, ou seja, uma redução de 9,9% na taxa de rejeição. Como resultado, a tarefa de atualização do modelo com base nas instâncias rejeitadas pode ajustar o modelo ao longo do tempo. No entanto, realizando atualizações do modelo por meio de uma abordagem FL, as organizações podem executar essa tarefa com mais

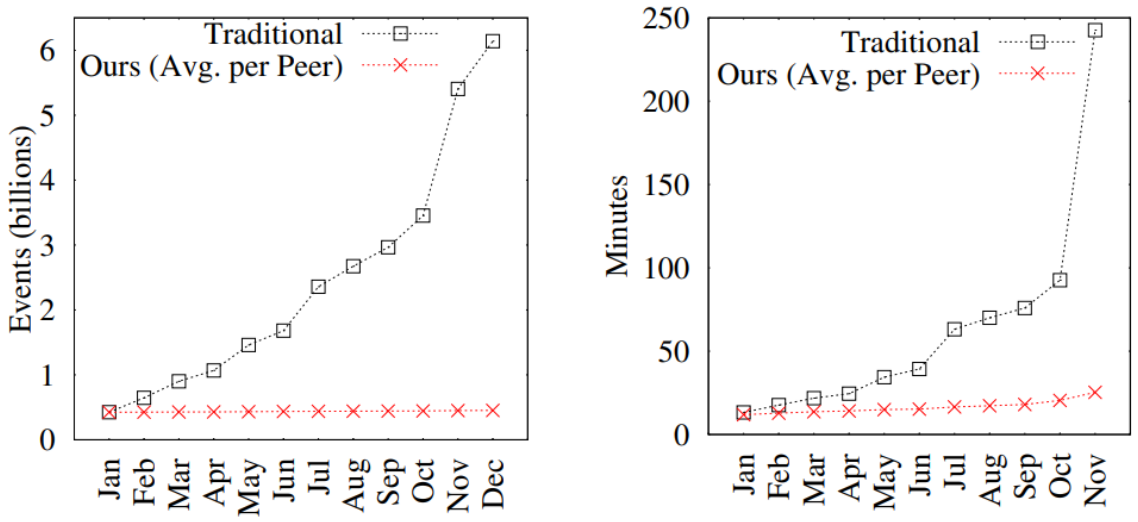
frequência, uma vez que os participantes coletarão mais tráfego de rede e compartilharão suas CTIs.



A) Sem Atualizações

B) Com Atualizações

Figura 19 -. Comportamento de precisão de técnicas selecionadas no conjunto de dados MAWIFlow.



C) Número acumulado de eventos para treinamento.

D) Custos cumulativos de processamento para treinamento.

Figura 20 - Comparação operacional do modelo proposto com técnicas tradicionais em termos de eventos utilizados e custos de processamento durante a fase de treinamento.

Além disso, foi investigado como a abordagem se comporta em comparação com técnicas tradicionais. A Figura 19 mostra a Medida F ao longo do tempo da proposta em comparação com as abordagens tradicionais. O modelo proposto melhora a precisão de classificação com e sem considerar as atualizações do modelo. Em média, em um cenário sem atualização (Figura 19-A), a proposta fornece uma Medida F de 0,89, um aumento de 0,23 e 0,13 em comparação com o RF e kNN, respectivamente. Supondo o uso das

atualizações do modelo (Figura 19-B), alcançamos uma Medida F média de 0,96, melhorando-a em 0,03 e 0,04 em comparação com o RF e kNN, respectivamente.

A principal vantagem da abordagem proposta, diz respeito à quantidade de dados que devem ser rotulados para realizar as atualizações do modelo à medida que o tempo passa. A Figura 20-A mostra o número acumulado de eventos de rede que devem ser rotulados à medida que o tempo passa para cada técnica selecionada. A abordagem proposta requer uma média de apenas 6,5% dos eventos em comparação com a abordagem tradicional (Proposta FL vs. FL Tradicional). Considerando os eventos necessários após o treinamento inicial em janeiro, a proposta melhora significativamente a abordagem tradicional, exigindo uma média de apenas 0,3% dos eventos a serem rotulados à medida que o tempo passa. No entanto, como as organizações podem compartilhar suas CTIs devido à aplicação de FL, podemos reduzir ainda mais os dados de treinamento necessários de acordo com o número de pares disponíveis.

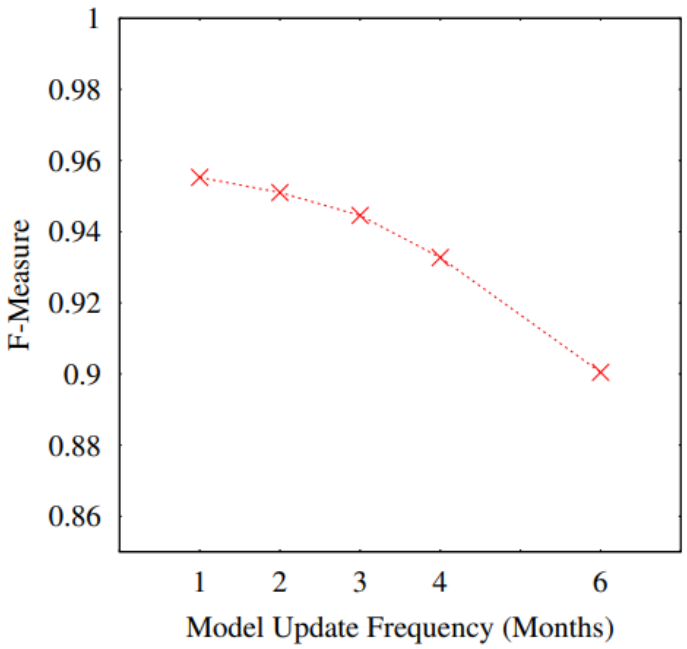


Figura 21 - Desempenho proposto do modelo de acordo com a vida útil do modelo, ou seja, frequência de atualizações do modelo.

Foi investigado ainda mais os custos de processamento de treinamento da solução proposta em comparação com as técnicas tradicionais. Para alcançar esse objetivo, foi medido o tempo de treinamento conforme o tempo passa usado por nossa proposta com o verificador e as atualizações mensais (Figura 18). Os experimentos foram executados

em um cluster de 10 nós, cada um equipado com uma CPU Intel i7 de 8 núcleos, 16GB de memória, interconectados por uma rede gigabit, executando em um sistema operacional Ubuntu v.22.04. A técnica Ensemble tradicional usa toda a capacidade de processamento do nó, enquanto o modelo proposto usa todo o processamento do cluster quando disponível. A Figura 20 mostra os custos acumulados de processamento de treinamento de nossa solução em as demais técnicas. A abordagem proposta reduz significativamente os custos de processamento em comparação com a técnica tradicional. A proposta requer em média apenas 10% dos custos de processamento por nó em comparação com a abordagem Ensemble tradicional. Isso ocorre porque a proposta pode selecionar proativamente quais eventos devem ser usados para as atualizações do modelo (Figura 20 - A), facilitando significativamente os custos de processamento para as atualizações do modelo à medida que o tempo passa.

Por fim, foi investigado como a abordagem proposta pode relaxar a frequência com que as atualizações do modelo são realizadas. Ao variar a frequência de realização da atualização do modelo com base nas instâncias rejeitadas para alcançar esse objetivo. A Figura 21 mostra o desempenho de classificação de acordo com a periodicidade da atualização do modelo em nossa proposta. A abordagem proposta não é significativamente afetada por um tempo de vida do modelo mais longo, com um impacto marginal na precisão do sistema. Por exemplo, o modelo atualizado mensalmente fornece uma Medida F média de 0,96, enquanto seu equivalente atualizado a cada semestre atinge 0,90, um impacto de apenas 0,06. Um tempo de vida mais longo do modelo beneficia significativamente uma implantação operacional realista do NIDS baseado em ML. Isso ocorre porque o operador da rede pode usar seu modelo de ML construído por períodos mais longos antes de realizar uma atualização do modelo, e quando isso ocorrer, menos eventos serão necessários e menos custos de processamento serão exigidos.

6.2.3. Considerações Finais

A abordagem de atualização de modelos em sistemas de detecção de intrusões baseados em aprendizado de máquina utilizando aprendizado federado (FL) oferece diversas vantagens em comparação com técnicas tradicionais. O FL tradicional, sem atualizações periódicas, apresenta uma queda significativa na precisão ao lidar com novos dados de tráfego de rede ao longo do tempo, como demonstrado por uma taxa crescente de falsos positivos. A técnica de rejeição proposta melhora a confiabilidade da

classificação, reduzindo a taxa de erro ao filtrar eventos de baixa confiança, o que é crucial para manter a precisão mesmo sem atualizações constantes do modelo.

Além disso, o modelo proposto com atualizações periódicas com base em eventos rejeitados mostra uma melhoria significativa na precisão e na redução de eventos rejeitados, comparado ao FL tradicional. A abordagem não apenas ajusta o modelo ao longo do tempo, mas também reduz a necessidade de rotulação de grandes volumes de dados, o que é eficiente em termos de custos de processamento. Este método permite que as organizações mantenham seus sistemas de detecção de intrusões atualizados e precisos, colaborando sem comprometer a privacidade dos dados. Em resumo, a integração de técnicas de rejeição e atualizações periódicas em um framework de FL promove uma manutenção contínua e eficiente da eficácia do sistema de detecção de intrusões, essencial para enfrentar as crescentes ameaças cibernéticas.

Capítulo 7

Conclusão

O crescente aumento dos ataques cibernéticos e a dinâmica do tráfego de rede impõem desafios significativos à eficácia dos Sistemas de Detecção de Intrusão (NIDS) baseados em técnicas tradicionais de aprendizado de máquina (ML). Em ambientes de produção, esses sistemas frequentemente demonstram uma perda de desempenho ao longo do tempo, uma vez que o tráfego de rede é caracterizado por sua natureza não estacionária, alterando-se com a introdução de novos serviços, ataques e padrões de comportamento. Este fenômeno compromete a precisão e a confiabilidade das abordagens tradicionais, que frequentemente assumem um comportamento estático no tráfego de rede.

Este trabalho se propôs a abordar essa limitação ao explorar o uso do aprendizado por reforço (RL), especificamente o algoritmo Q-Learning, como uma solução para melhorar a longevidade e a precisão dos modelos de detecção de intrusão. Ao contrário das abordagens tradicionais, que dependem de atualizações periódicas e rotulagem manual, o modelo proposto através de RL mostrou capacidade de adaptação contínua ao comportamento dinâmico da rede, mantendo taxas de falsos positivos (FP) e falsos negativos (FN) dentro de limites aceitáveis, mesmo sem intervenções frequentes.

Além disso, ao considerar a atualização semestral do modelo, a proposta demonstrou que a implementação de um sistema de aprendizado por reforço pode reduzir significativamente os custos associados à atualização de modelos em ambientes de produção. Ao contrário dos métodos convencionais, que necessitam de atualizações frequentes, o modelo de RL foi capaz de convergir mais rapidamente, aproveitando o conhecimento adquirido durante períodos de desatualização. Essa característica representa uma solução vantajosa, principalmente em cenários de produção onde a rotulagem de grandes volumes de eventos de rede é um processo oneroso e complexo.

Os experimentos realizados com o conjunto de dados MAWIFlow, abrangendo quatro anos de tráfego de rede, reforçaram a constatação de que as abordagens tradicionais não

conseguem manter a precisão ao longo do tempo sem atualizações periódicas. As taxas de FP e FN apresentaram variações significativas com o passar dos meses, enquanto o modelo baseado em RL mostrou uma variação significativamente menor, garantindo maior robustez em relação às flutuações naturais do tráfego de rede. Estes resultados indicam que o aprendizado por reforço pode fornecer uma solução eficiente para a detecção de intrusão em cenários de produção dinâmicos, onde a atualização contínua dos modelos é desafiadora.

Além do aprendizado por reforço, o trabalho também explorou a viabilidade do aprendizado federado (FL) como uma abordagem complementar para a atualização descentralizada e colaborativa dos modelos de detecção de intrusão. A utilização conjunta de RL e FL possibilitou o desenvolvimento de um sistema de NIDS mais escalável e adaptativo, capaz de realizar atualizações contínuas utilizando dados distribuídos sem comprometer a privacidade. A implementação dessa estratégia colaborativa demonstrou sua eficácia na adaptação contínua dos modelos às mudanças no tráfego de rede, proporcionando uma maior resiliência às ameaças emergentes e garantindo a integridade dos dados.

A combinação de aprendizado por reforço com aprendizado federado representa uma proposta inovadora que supera as limitações das abordagens tradicionais. A integração dessas técnicas oferece um sistema de detecção de intrusão capaz de se adaptar de forma autônoma às mudanças no ambiente de rede, sem a necessidade de intervenções manuais contínuas. Além disso, a estratégia adaptativa de classificação implementada no modelo permite uma maior confiabilidade mesmo diante de padrões de tráfego desconhecidos, o que reforça a robustez e a eficácia do sistema em um cenário de ameaças cibernéticas dinâmicas.

Desta forma, este trabalho propõe uma solução que não apenas resolve problemas críticos de longevidade, precisão e atualização de modelos em sistemas de detecção de intrusão, mas também estabelece um novo paradigma para a segurança cibernética em ambientes de rede dinâmicos. A abordagem apresentada contribui para o avanço das metodologias de detecção de intrusão baseadas em aprendizado de máquina, oferecendo uma solução

mais escalável, eficiente e confiável, capaz de enfrentar os desafios impostos por ataques cibernéticos cada vez mais sofisticados e complexos.

Referências

[1] Kaspersky Security Bulletin 2022. Statistics, 2023. [Online]. Available: https://securelist.com/ksb-2022-statistics/108129/

[2] Kaspersky Lab. “Kaspersky Security Bulletin 2019. Statistics,” 2019. [Online]. Available: https://securelist.com/kaspersky-security-bulletin2019-statistics/95475/

[3] B. Molina-Coronado, U. Mori, A. Mendiburu, and J. Miguel-Alonso, “Survey of Network Intrusion Detection Methods From the Perspective of the Knowledge Discovery in Databases Process,” IEEE Trans. on Network and Service Management, vol. 17, no. 4, pp. 2451–2479, Dec. 2020.

[4] E. Viegas, A. Santin, A. Bessani, and N. Neves, “BigFlow: Real-time and reliable anomaly-based intrusion detection for high-speed networks,” Future Generation Computer Systems, vol. 93, pp. 473–485, Apr. 2019.

[5] T. Zoppi, A. Ceccarelli, T. Puccetti, and A. Bondavalli, “Which algorithm can detect unknown attacks? comparison of supervised, unsupervised and meta-learning algorithms for intrusion detection,” Computers & Security, vol. 127, p. 103107, 2023.

[6] R. Sommer and V. Paxson, “Outside the closed world: On using machine learning for network intrusion detection,” in Proc. IEEE Symp. Security Privacy, 2010, pp. 305–316. [Online]. Available: https://doi.org/10.1109/sp.2010.25

[7] M. Ramkumar, P. B. Reddy, J. Thirukrishna, and C. Vidyadhari, “Intrusion detection in big data using hybrid feature fusion and optimization enabled deep learning based on spark architecture,” Computers & Security, vol. 116, p. 102668, 2022.

[8] Y. Al-Hadhrami and F. K. Hussain, “Real time dataset generation framework for intrusion detection systems in IoT,” Future Generation Computer Systems, vol. 108, pp. 414–423, Jul. 2020.

[9] I. F. Kilincer, F. Ertam, and A. Sengur, “Machine learning methods for cyber security intrusion detection: Datasets and comparative study,” Computer Networks, vol. 188, p. 107840, 2021.

[10] B. Molina-Coronado, U. Mori, A. Mendiburu, and J. Miguel-Alonso, “Survey of network intrusion detection methods from the perspective of the knowledge discovery in databases process,” IEEE Trans. Netw. Service Manag., vol. 17, no. 4, pp. 2451–2479, Dec. 2020.

[11] R. Mills, A. K. Marnerides, M. Broadbent, and N. Race, “Practical Intrusion Detection of Emerging Threats,” IEEE Transactions on Network and Service Management, vol. 19, no. 1, pp. 582–600, Mar. 2022.

[12] B. Li, Y. Wang, K. Xu, L. Cheng, and Z. Qin, “DFAID: Densityaware and feature-deviated active intrusion detection over network traffic streams,” Computers & Security, vol. 118, p. 102719, 2022.

[13] A. Thakkar and R. Lohiya, “A survey on intrusion detection system: feature selection, model, performance measures, application perspective, challenges, and future research directions,” Artificial Intelligence Review, vol. 55, no. 1, pp. 453–563, Jul. 2021.

[14] X. Li, Z. Hu, M. Xu, Y. Wang, and J. Ma, “Transfer learning based intrusion detection scheme for Internet of vehicles,” Information Sciences, vol. 547, pp. 119–135, 2021.

[15] J.-S. Lee, Y.-C. Chen, C.-J. Chew, C.-L. Chen, T.-N. Huynh, and C.W. Kuo, “Conn-ids: Intrusion detection system based on collaborative neural networks and agile training,” Computers & Security, vol. 122, p. 102908, 2022.

[16] Daniel Arp, Erwin Quiring, Feargus Pendlebury, Alexander Warnecke, F. Pierazzi, Christian Wressnegger, L. Cavallaro, and Konrad Rieck, Dos and Don’ts of Machine Learning in Computer Security, 2022nd ed., ser. Usenix Security Symposium (USENIX). USENIX, Jul. 2021.

[17] D. Arp et al., “DoS and dont’s of machine learning in computer security,” in Proc. 31st USENIX Security Symp. (USENIX Security), Aug. 2022, pp. 3971–3988. [Online]. Available: https://www.usenix.org/conference/usenixsecurity22/presentation/arp.

[18] C. Gates and C. Taylor, “Challenging the anomaly detection paradigm: A provocative discussion,” in Proc. Workshop New Security Paradigms (NSPW), 2006, pp. 21–29.

[19] C. Zhang, D. Jia, L. Wang, W. Wang, F. Liu, and A. Yang, “Comparative research on network intrusion detection methods based on machine learning,” Computers & Security, vol. 121, p. 102861, 2022.

[20] Alpaydin, E. (2020). Introduction to Machine Learning. MIT Press.

[21] Bishop, C. M. (2006). Pattern Recognition and Machine Learning. Springer.

[22] Chapelle, O., Scholkopf, B., & Zien, A. (2006). Semi-Supervised Learning. MIT Press.

[23] Domingos, P. (2015). The Master Algorithm: How the Quest for the Ultimate Learning Machine Will Remake Our World. Basic Books.

[24] Goodfellow, I., Bengio, Y., & Courville, A. (2016). Deep Learning. MIT Press.

[25] Hastie, T., Tibshirani, R., & Friedman, J. (2009). The Elements of Statistical Learning: Data Mining, Inference, and Prediction. Springer.

[26] Mitchell, T. M. (1997). Machine Learning. McGraw-Hill.

[27] Murphy, K. P. (2012). Machine Learning: A Probabilistic Perspective. MIT Press.

[28] Mnih, V., et al. (2015). "Human-level control through deep reinforcement learning." Nature.

[29] Sutton, R. S., & Barto, A. G. (2018). Reinforcement Learning: An Introduction. MIT Press.

[30] Disha, R.A., Waheed, S. Performance analysis of machine learning models for intrusion detection system using Gini Impurity-based Weighted Random Forest (GIWRF) feature selection technique. Cybersecurity 5, 1 (2022). https://doi.org/10.1186/s42400-021-00103-8.

[31] Zagajewski, B.; Kluczek, M.; Raczko, E.; Njegovec, A.; Dabija, A.; Kycko, M. Comparison of Random Forest, Support Vector Machines, and Neural Networks for Post-Disaster Forest Species Mapping of the Krkonoše/Karkonosze Transboundary Biosphere Reserve. Remote Sens. 2021, 13, 2581. https://doi.org/10.3390/rs13132581.

[32] Z. Yang, X. Liu, T. Li, D. Wu, J. Wang, Y. Zhao, and H. Han, “A systematic literature review of methods and datasets for anomaly-based network intrusion detection,” Computers & Security, vol. 116, p. 102675, 2022.

[33] Z. K. Maseer, R. Yusof, N. Bahaman, S. A. Mostafa, and C. F. M. Foozy, “Benchmarking of Machine Learning for Anomaly Based Intrusion Detection Systems in the CICIDS2017 Dataset,” IEEE Access, vol. 9, pp. 22351–22370, 2021.

[34] E. Papadogiannaki and S. Ioannidis, “A Survey on Encrypted Network Traffic Analysis Applications, Techniques, and Countermeasures,” ACM Comput. Surv., vol. 54, no. 6, jul 2021.

[35] M. M. Yamin, B. Katt, and V. Gkioulos, “Cyber ranges and security testbeds: Scenarios, functions, tools and architecture,” Computers & Security, vol. 88, p. 101636, Jan. 2020.

[36] R. Fontugne, P. Borgnat, P. Abry, and K. Fukuda, “MAWILab: Combining Diverse Anomaly Detectors for Automated Anomaly Labeling and Performance Benchmarking,” in Proc. of the 6th Int. Conf. on emerging Networking EXperiments and Technologies (CoNEXT), 2010.

[37] R. Sommer and V. Paxson, “Outside the Closed World: On Using Machine Learning for Network Intrusion Detection,” in 2010 IEEE Symposium on Security and Privacy. IEEE, 2010.

[38] Cortes, C., & Vapnik, V. (1995). Support-vector networks. Machine Learning, 20(3), 273-297.

[39] Breiman, L. (2001). Random forests. Machine Learning, 45(1), 5-32.

[40] LeCun, Y., Bengio, Y., & Hinton, G. (2015). Deep learning. Nature, 521(7553), 436-444.

[41] Schmidhuber, J. (2015). Deep learning in neural networks: An overview. Neural Networks, 61, 85-117.

[42] Breiman, L. (2001). Random forests. Machine Learning, 45(1), 5-32.

[43] Cover, T., & Hart, P. (1967). Nearest neighbor pattern classification. IEEE Transactions on Information Theory, 13(1), 21-27.

[44] Breiman, L. (1996). Bagging predictors. Machine Learning, 24(2), 123-140.

[45] Kuncheva, L. I. (2004). Combining pattern classifiers: Methods and algorithms. John Wiley & Sons.

[46] S. Wassermann, T. Cuvelier, P. Mulinka, and P. Casas, “Adaptive and reinforcement learning approaches for online network monitoring and analysis,” IEEE Trans. Netw. Service Manag., vol. 18, no. 2, pp. 1832–1849, Jun. 2021.

[47] M. Lopez-Martin, B. Carro, and A. Sanchez-Esguevillas, “Application of deep reinforcement learning to intrusion detection for supervised problems,” Exp. Syst. Appl., vol. 141, Mar. 2020, Art. no. 112963. [Online]. Available: https://doi.org/10.1016/j.eswa.2019.112963.

[48] McMahan, H. Brendan, et al. "Communication-efficient learning of deep networks from decentralized data." Aistats. Vol. 1. No. 2. 2017.

[49] Yang, Qiang, et al. "Federated machine learning: Concept and applications." ACM Transactions on Intelligent Systems and Technology (TIST) 10.2 (2019): 1-19.

[50] Sharma, Pranshu, e Rajesh Kumar. "Federated Learning: Strategies for Improving Communication Efficiency."

[51] X. Hei, X. Yin, Y. Wang, J. Ren, and L. Zhu, “A trusted feature aggregator federated learning for distributed malicious attack detection,” Computers & Security, vol. 99, p. 102033, 2020.

[52] R. Alghamdi and M. Bellaiche, “A cascaded federated deep learning-based framework for detecting wormhole attacks in iot networks,” Computers & Security, vol. 125, p. 103014, 2023.

[53] Y. Zhou, Q. Ye, and J. Lv, “Communication-efficient federated learning with compensated overlap-fedavg,” IEEE Transactions on Parallel and Distributed Systems, vol. 33, no. 1, pp. 192–205, 2021.

[54] T. Saba, A. Rehman, T. Sadad, H. Kolivand, and S. A. Bahaj, “Anomalybased intrusion detection system for iot networks through deep learning model,” Computers and Electrical Engineering, vol. 99, p. 107810, 2022.

[55] Z. Zeng, W. Peng, and D. Zeng, “Improving the Stability of Intrusion Detection With Causal Deep Learning,” IEEE Transactions on Network and Service Management, vol. 19, no. 4, pp. 4750–4763, Dec. 2022.

[56] Q. Zhao, M. Chen, Z. Gu, S. Luan, H. Zeng, and S. Chakrabory, “CAN Bus Intrusion Detection Based on Auxiliary Classifier GAN and Outof-distribution Detection,” ACM Transactions on Embedded Computing Systems, vol. 21, no. 4, pp. 1–30, Jul. 2022.

[57] O. A. Wahab, “Intrusion Detection in the IoT Under Data and Concept Drifts: Online Deep Learning Approach,” IEEE Internet of Things Journal, vol. 9, no. 20, pp. 19706–19716, Oct. 2022.

[58] A. Ahmim, L. Maglaras, M. A. Ferrag, M. Derdour, and H. Janicke, “A novel hierarchical intrusion detection system based on decision tree and rules-based models,” in 2019 15th International Conference on Distributed Computing in Sensor Systems (DCOSS). IEEE, May 2019.

[59] J. Kevric, S. Jukic, and A. Subasi, “An effective combining classifier approach using tree algorithms for network intrusion detection,” Neural Computing and Applications, vol. 28, no. S1, pp. 1051–1058, Jun. 2016.

[60] R. Vinayakumar, M. Alazab, K. P. Soman, P. Poornachandran, A. AlNemrat, and S. Venkatraman, “Deep learning approach for intelligent intrusion detection system,” IEEE Access, vol. 7, pp. 525–550, 2019.

[61] H. Yang and F. Wang, “Wireless network intrusion detection based on improved convolutional neural network,” IEEE Access, vol. 7, pp. 64366–64374, 2019.

[62] X. Li, W. Chen, Q. Zhang, and L. Wu, “Building auto-encoder intrusion detection system based on random forest feature selection,” Computers & Security, vol. 95, p. 101851, Aug. 2020.

[63] Y. Yan, L. Qi, J. Wang, Y. Lin, and L. Chen, “A network intrusion detection method based on stacked autoencoder and LSTM,” in ICC 2020 IEEE Int. Conf. on Communications (ICC). IEEE, Jun. 2020.

[64] E. Mahdavi, A. Fanian, A. Mirzaei, and Z. Taghiyarrenani, “ITLIDS: Incremental Transfer Learning for Intrusion Detection Systems,” Knowledge-Based Systems, vol. 253, p. 109542, 2022.

[65] Y. Huang and M. Ma, “ILL-IDS: An incremental lifetime learning IDS for VANETs,” Computers & Security, vol. 124, p. 102992, 2023.

[66] J. Jiang, F. Liu, W. W. Y. Ng, Q. Tang, W. Wang, and Q.-V. Pham, “Dynamic Incremental Ensemble Fuzzy Classifier for Data Streams in Green Internet of Things,” IEEE Transactions on Green Communications and Networking, vol. 6, no. 3, pp. 1316–1329, 2022.

[67] Z. Wu, P. Gao, L. Cui, and J. Chen, “An Incremental Learning Method Based on Dynamic Ensemble RVM for Intrusion Detection,” IEEE Transactions on Network and Service Management, vol. 19, no. 1, pp. 671–685, 2022.

[68] Suwannalai and C. Polprasert, “Network intrusion detection systems using adversarial reinforcement learning with deep Q-network,” in Proc. 18th Int. Conf. ICT Knowl. Eng. (ICT KE), Nov. 2020, pp. 1–9. [Online]. Available: https://doi.org/10.1109/ictke50349.2020.9289884

[69] H. Benaddi, K. Ibrahimi, A. Benslimane, and J. Qadir, A Deep Reinforcement Learning Based Intrusion Detection System (DRL-IDS) for Securing Wireless Sensor Networks and Internet of Things (Lecture Notes of the Institute for Computer Sciences, Social Informatics and Telecommunications Engineering). Cham, Switzerland: Springer Int., 2020, pp. 73–87. [Online]. Available: https://doi.org/10.1007/978-3-030-52988-8_7.

[70] A. Servin and D. Kudenko, “Multi-agent reinforcement learning for intrusion detection,” in Adaptive Agents and Multi-Agent Systems III. Adaptation and Multi-Agent Learning. Heidelberg, Germany: Springer, 2008, pp. 211–223. [Online]. Available: https://doi.org/10.1007/978-3-540-77949-0_15.

[71] S. Yuan, H. Li, R. Zhang, M. Hao, Y. Li, and R. Lu, “Towards Lightweight and Efficient Distributed Intrusion Detection Framework,” in 2021 IEEE Global Communications Conference (GLOBECOM). IEEE, Dec. 2021.

[72] Y. Sun, H. Ochiai, and H. Esaki, “Intrusion Detection with Segmented Federated Learning for Large-Scale Multiple LANs,” in 2020 International Joint Conference on Neural Networks (IJCNN). IEEE, Jul. 2020.

[73] V. Mothukuri, P. Khare, R. M. Parizi, S. Pouriyeh, A. Dehghantanha, and G. Srivastava, “Federated-Learning-Based Anomaly Detection for IoT Security Attacks,” IEEE Internet of Things Journal, vol. 9, no. 4, pp. 2545–2554, Feb. 2022.

[74] X. Hei, X. Yin, Y. Wang, J. Ren, and L. Zhu, “A trusted feature aggregator federated learning for distributed malicious attack detection,” Computers & Security, vol. 99, p. 102033, Dec. 2020.

[75] scikit-learn Machine Learning in Python, 2023. [Online]. Available: https://scikit-learn.org/