Detecção de Intrusão Através de Redes Neurais Profundas com Saídas Antecipadas para Inferência Rápida e Confiável

João André Simioni[1], Eduardo Kugler Viegas[1], Altair Olivo Santin[1], Pedro Horchulhack[1]

[1]Programa de Pós-Graduação em Informática (PPGIa)
Pontifícia Universidade Católica do Paraná (PUCPR)
80.215-901 – Curitiba – PR

{joao.asimioni, eduardo.viegas, santin, pedro.horchulhack}@ppgia.pucpr.br

Abstract. *Deep Neural Networks (DNN) are the state-of-the-art in intrusion detection, but they increase computational costs and are usually impractical for resource-limited devices. We present a novel early-exit DNN for fast inference and reliable intrusion detection. Our approach divides the DNN into branches, classifying most samples into the initial branches to reduce inference costs. Challenging samples in the latter branch are classified with a reject option, improving reliability. The multi-objective optimized method was tested on an 8 TB data set, reducing computational costs by up to 82% and decreasing error rates by up to 3.3%.*

Resumo. *Redes Neurais Profundas (DNN) representam o estado da arte em detecção de intrusões, mas aumentam os custos computacionais, tornando-as impraticáveis para dispositivos com recursos limitados. Apresentamos uma nova DNN com saída antecipada para inferência rápida e detecção confiável de intrusões. Nossa abordagem divide a DNN em ramificações, classificando a maioria das amostras nas ramificações iniciais para reduzir os custos de inferência. Amostras que apresentam um desafio maior de classificação são categorizadas na última ramificação, utilizando uma opção de rejeição para aprimorar a confiabilidade. O método de otimização multi-objetivo foi testado em um conjunto de dados de 8 TB, resultando em uma redução de até 82% nos custos computacionais e diminuição das taxas de erro em até 3,3%.*

1. Introdução

A utilização de dispositivos com recursos limitados, tais como os usados no contexto de Internet of Things (IoT), aumentou significativamente nos últimos anos. Esses dispositivos são geralmente sistemas computacionais embarcados alimentados por bateria, com capacidade limitada de processamento e normalmente operam conectados a uma rede. Devido à sua ampla utilização, esses dispositivos são os principais alvos de ataques cibernéticos, com relatórios indicando um aumento de 40% nos ataques apenas no último ano [SonicWall 2023, Abreu et al. 2017].

Para proteger dispositivos de IoT, uma abordagem comum é a utilização de *Network-based Intrusion Detection System (NIDS)*, que monitoram o tráfego de rede dos

dispositivos identificando padrões suspeitos [Viegas et al. 2017]. Com o passar dos anos, muitas técnicas foram propostas com esse objetivo, sendo elas normalmente categorizadas como *baseadas em assinatura* ou *baseadas em comportamento* [Al-Garadi et al. 2020]. Por um lado, a abordagem *baseada em assinatura* identifica tentativas de intrusão comparando o tráfego de rede com um banco de dados de ações maliciosas bem conhecidas, limitando sua habilidade de detecção apenas a ataques conhecidos previamente. Por outro lado, técnicas *baseada em comportamento* identificam irregularidades com base em desvios de um perfil previamente estabelecido, sendo potencialmente capaz de detectar tipos de ataque desconhecidos.

Buscando esse objetivo, muitos trabalhos propõem novas técnicas de detecção de intrusão *baseada em comportamento*, nos quais as abordagens baseadas em *Deep Neural Networks (DNN)* frequentemente atingem os maiores níveis de acurácia [Ma et al. 2020]. Para atingir tal objetivo, pesquisadores, em sua vasta maioria, aumentam o número de parâmetros de suas arquiteturas de DNN procurando melhores acurácias. Essa abordagem, além de aumentar os custos computacionais da inferência, também torna esses modelos inadequados para dispositivos com recursos limitados, devido à restrição de memória e baixa capacidade de processamento [Ge et al. 2021].

Ao ser utilizada, a detecção de intrusão baseada em DNN implica em processar os parâmetros de entrada por toda a arquitetura da rede, até que a camada de saída seja alcançada. Nesse processo, múltiplas características e padrões não lineares são extraídos, servindo então como indicadores para a tarefa de classificação realizada na camada de saída. Dessa forma, independentemente da complexidade do evento avaliado, seja ele simples ou complexo, a decisão somente pode ser tomada uma vez que todos os indicadores forem extraídos, potencialmente resultando em um uso ineficiente dos recursos computacionais [Li et al. 2020]. Levando isso em consideração, nos últimos anos, alguns estudos exploraram a introdução de saídas antecipadas em arquiteturas DNN. A técnica de saídas antecipadas adiciona múltiplos pontos de terminação que dividem a rede em ramificações, permitindo que a tarefa de inferência seja concluída antecipadamente se os padrões e características já extraídas puderem, de forma confiável, obter uma decisão [Seifeddine et al. 2021].

Enquanto as saídas antecipadas tipicamente reduzem o custo computacional de inferência das DNNs com impacto mínimo na acurácia, elas não podem ser imediatamente aplicadas à detecção de intrusão de rede, especialmente em dispositivos com recursos limitados [Laskaridis et al. 2021]. Em contraste com outras áreas, em relação aos dados analisados, o comportamento do tráfego de rede é notavelmente dinâmico e evolui continuamente. Isso exige capacidades avançadas de generalização do modelo da DNN, enquanto o dinamismo do comportamento geralmente só pode ser corrigido por meio de atualizações do modelo [Fontugne et al. 2010]. Por outro lado, estratégias existentes de detecção de intrusão baseadas em saídas antecipadas frequentemente deixam de considerar as compensações na generalização do modelo que surgem ao encerrar antecipadamente a tarefa de inferência. Além disso, geralmente é possível corrigir os resultados não confiáveis de classificação de um modelo DNN desatualizado, devido à mudança no comportamento de tráfego de rede, através da atualização do modelo. Porém, esse procedimento frequentemente necessita de um longo período, muitas vezes por dias ou semanas, obrigando o modelo implantado a ter uma vida útil prolongada para garantir sua

confiabilidade.

Contribuição. Diante disso, este artigo apresenta uma nova abordagem utilizando DNN com saídas antecipadas, visando reduzir tempos de inferência ao mesmo tempo que mantém a confiabilidade da detecção de intrusão. O modelo proposto foi implementado usando duas estratégias. Primeiro, tratamos as saídas antecipadas na detecção de intrusão usando DNN como uma tarefa de otimização multi-objetivo. Nosso objetivo é reduzir o tempo de inferência do modelo e aumentar a acurácia do sistema de detecção por meio da otimização da seleção dos limiares de saída. Após isso, nós introduzimos uma abordagem de classificação com uma opção de rejeição na última ramificação da DNN, permitindo a rejeição de classificações potencialmente não confiáveis, influenciadas pelo novo comportamento de tráfego de rede. Como resultado, nós garantimos a confiabilidade da classificação através da opção de rejeição ao mesmo tempo que reduzimos os custos computacionais da inferência ao aplicar as saídas antecipadas.

Em resumo, as principais contribuições do artigo são:

- Uma avaliação da confiabilidade da classificação de arquiteturas de DNN amplamente usadas para tarefas de detecção de intrusão. Experimentos em um conjunto de dados contendo um ano de tráfego de rede real demonstraram que as abordagens atuais demandam processamento computacional impraticável e também sofrem de uma redução na acurácia nos meses subsequentes ao treinamento.
- Uma nova nova abordagem com DNN com saídas antecipadas para detecção de intrusão confiável implementada através de otimização multi-objetivo associada a uma classificação com a opção de rejeição. Nossa proposta pode reduzir a taxa de erro média em até 3.3 enquanto reduz o custo computacional de inferência por até 82%.

2. Fundamentação

2.1. Sistema de detecção de intrusão baseado em rede para IoT

NIDSs *baseados em comportamento* têm sido amplamente utilizados por operadores de rede para monitorar a comunicação entre dispositivos e detectar atividades maliciosas [Geremias et al. 2022]. Geralmente, essas ferramentas implementam detecção de intrusão baseado em DNN seguindo uma implementação modular sequencial, que abrange *Aquisição de Dados*, *Extração de Características*, *Classificação*, e *Alerta*. O primeiro módulo é responsável pela coleta em tempo real de pacotes de uma interface de rede específica. O comportamento dos dados coletados é extraído pelo módulo de *Extração de Características*, que tipicamente constrói um vetor, resumindo a comunicação entre as entidades de rede em uma janela de tempo específica [Horchulhack et al. 2024a]. O vetor resultante serve como entrada para o módulo de *Classificação*, que utiliza uma DNN pré-treinada para classificar os eventos como sendo *normal* ou *ataque*. Finalmente, o módulo *Alerta* envia eventos classificados como *ataque*.

A aplicação de DNNs para classificação de tráfego de rede em dispositivos com recursos limitados requer a prévia execução da etapa de treinamento [dos Santos et al. 2023]. O comportamento de um conjunto de dados de treinamento é extraído durante a fase de treinamento. Portanto, o conjunto de dados deveria, idealmente, possuir milhões de amostras de rede categorizadas que representam o que

é esperado em um ambiente de produção. A acurácia do modelo resultante é avaliada durante a fase de testes, e espera-se que as taxas de acurácia medidas indiquem sua performance em um ambiente de produção.

2.2. Saídas Antecipadas

Nos últimos anos, a acurácia das técnicas de detecção de intrusão propostas baseadas em DNN aumentou consistentemente [Ma et al. 2020]. Para abrir caminhos a detecções com maior acurácia, pesquisadores geralmente aumentam o número de parâmetros da DNN. Consequentemente, as soluções implementadas muitas vezes demandam custos computacionais impraticáveis para a inferência, inviabilizando suas aplicações em dispositivos com recursos limitados. Saídas antecipadas são projetadas para enfrentar esse desafio, introduzindo ramificações adicionais que permitem a terminação prematura da inferência da rede [Seifeddine et al. 2021]. Cada saída antecipada geralmente consiste em camadas totalmente conectadas, capazes de classificar a entrada na camada atual. Se uma amostra de entrada tiver confiança elevada em uma determinada ramificação, ela pode sair naquela ramificação sem percorrer toda a rede, reduzindo eficientemente a profundidade da rede que uma parcela das amostras precisa percorrer.

Para conduzir o processo de treinamento da DNN com saídas antecipadas, pesquisadores frequentemente utilizam o a técnica de treinamento conjunto [Al-Garadi et al. 2020]. Sendo N o número de saídas ramificadas e $\tilde{y}^i$ a saída de classificação de cada ramificação i de um dado evento com rótulo y. Podemos computar a função de perda conjunta como a soma ponderada das perdas de cada ramificação através da seguinte equação:

$$\mathcal{L}_{joint}(\tilde{y}^i, y^i) = \sum_{i=1}^{N} w_i \mathcal{L}(\tilde{y}^i, y^i) \tag{1}$$

onde $\mathcal{L}_{joint}$ é a função de perda conjunta, $\mathcal{L}$ a função de perda, e w_i o peso da ramificação i. Aqui, w_i pode ser usado para fazer o ajuste fino das acurácias em cada ramificação, de forma a permitir uma preferência em direção às saídas antecipadas com maior acurácia, resultando em uma redução geral no custo computacional.

Na fase de testes a tarefa de inferência da rede processa o evento de entrada até que ele encontre a primeira ramificação, prematuramente interrompendo a inferência com base na condição de que o limiar de classificação supere um limiar de confiança t. Tipicamente, o limiar de aceitação é estabelecido com base na decisão do operador da rede, considerando a relação desejada entre a acurácia e o tempo de inferência médio. Se a ramificação final é alcançada, a classificação do evento é determinada com base na decisão feita nessa ramificação.

3. Trabalhos Relacionados

NIDSs utilizando abordagens de DNN têm sido um assunto popular de pesquisa na literatura nos últimos anos [Molina-Coronado et al. 2020, Horchulhack et al. 2024b]. Geralmente, os esquemas propostos priorizam maiores acurácias na detecção, ignorando sua aplicabilidade prática. Por exemplo, Li Ma *et al.* [Ma et al. 2020] sugerem o uso de DNN para detecção de intrusão em IoT utilização a fusão de características. Apesar

dessa abordagem aumentar a acurácia da classificação em um conjunto de dados desatualizado, há uma tendência em ignorar os custos de inferência e problemas com a confiabilidade da classificação. Outra abordagem focada em acurácia foi proposta por J. Zhang *et al.* [Zhang et al. 2020], que recorre a um conjunto de DNN para a tarefa de detecção. Apesar dos benefícios na acurácia, a utilização de múltiplas DNNs prejudica sua utilização em dispositivos com recursos limitados. M. Ge *et al.* [Ge et al. 2021] propõe a aplicação de uma DNN para detectar intrusões enquanto considera os custos computacionais associados. Sua proposta superou métodos tradicionais simples mas negligenciou a confiabilidade na classificação.

Nos últimos anos, reconhecendo a impraticabilidade de aplicar esses métodos em dispositivos com recursos limitados, algumas pesquisas têm mudado o foco da acurácia para melhorar os custos computacionais da inferência [Al-Garadi et al. 2020, Santos et al. 2023]. Y. Wang *et al.* [Wang et al. 2023] propõe a aplicação da MobileNet, uma implementação leve de uma DNN, para detecção de intrusão. Os autores mostraram que altas acurácias podem ser alcançadas juntamente com altas taxas de detecção e requisitos de memória mínimos. No entanto, os autores negligenciaram o comportamento evolutivo do tráfego de rede. Enquanto a aplicação de saídas antecipadas tem sido extensivamente tratada em vários domínios [Laskaridis et al. 2021], sua adoção em detecção de intrusão ainda está em suas fases iniciais. En Li *et al.* [Li et al. 2020] propõe a aplicação de saídas antecipadas em detecção de intrusão para aumentar a taxa de detecção. Sua abordagem melhora o tempo de inferência de forma notável, ainda que não trate da influência das alterações do comportamento do tráfego de rede na confiabilidade de sua solução. Uma abordagem similar foi proposta por W. Seifeddine *et al.* [Seifeddine et al. 2021] que criou ramificações de inferência para distribuir a DNN em múltiplos dispositivos. Apesar dos autores conseguirem diminuir os custos computacionais da inferência, eles não atentaram para o efeito do comportamento não estacionário do tráfego de rede em seu modelo. Como resultado, há uma lacuna na literatura para a detecção de intrusão que considere o comportamento evolutivo do tráfego de rede enquanto também endereça os custos computacionais da inferência.

4. DNN com Saídas Antecipadas para Inferência Rápida e Detecção de Intrusão Confiável

Para abordar o comportamento evolutivo do tráfego de rede ao mesmo tempo em que se minimizam as demandas de processamento, nós introduzimos um modelo de detecção de intrusão realizado através de saídas antecipadas com uma opção de rejeição. A visão geral do modelo é ilustrada na Figura 1 e é implementado segundo duas etapas principais, *Módulo de Rejeição*, e *Otimização Multi-objetivo*.

O objetivo do *Módulo de Rejeição* é identificar classificações não confiáveis na última ramificação da DNN, mesmo se desatualizada. Classificações não confiáveis são atribuídas aos novos comportamentos de tráfego de rede que podem causar um aumento nas taxas de classificações incorretas. Como resultado, diante do longo prazo necessário para obter um modelo atualizado, nós utilizamos a classificação com a opção de rejeição para estender a vida útil do modelo. Nós assumimos que a confiança da classificação pode servir como uma medida da qualidade da classificação, e portanto, a rejeição de classificações com baixa confiança pode ajudar a aumentar de forma eficiente a confiabilidade do sistema.

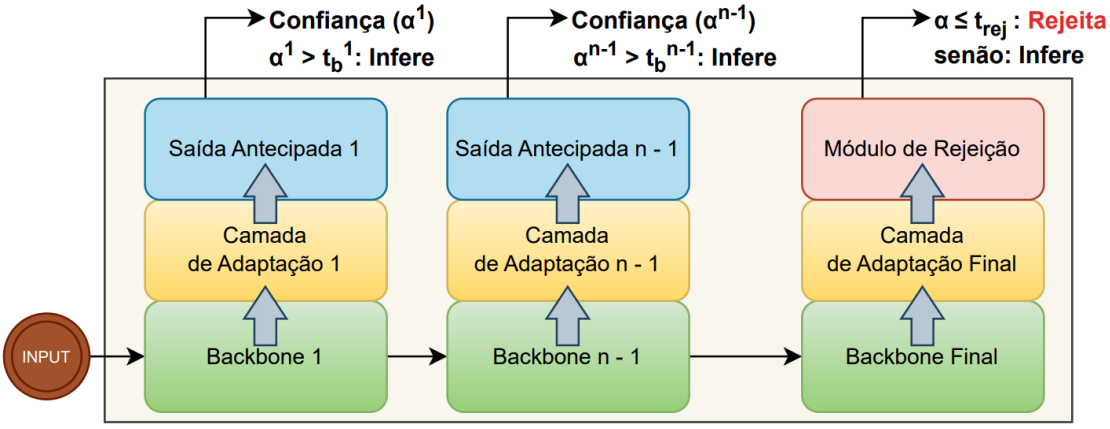

Figura 1. Visão geral do modelo proposto para detecção de intrusão confiável baseado em saídas antecipadas para inferência rápida.

O objetivo da *Otimização Multi-objetivo* é pro-ativamente determinar o limiar de aceitação para as ramificações da DNN e o limiar de rejeição para a ramificação final (vide Figura 1, indicadas como limiares t_b e t_{rej}). Nossa principal ideia é otimizar os limiares das saídas antecipadas baseado no comportamento de tráfego de rede e, simultaneamente, aumentar a confiabilidade do sistema através da classificação com a opção de rejeição. Como resultado, nosso modelo pode reduzir os custos de computação associados à inferência enquanto garante a confiabilidade do sistema.

A subseções seguintes apresentam uma descrição detalhada do modelo proposto.

4.1. Classificação com Opção de Rejeição

O comportamento do tráfego de rede é altamente variável e também evolui com o passar do tempo. Considerando a natureza não estacionária do comportamento do tráfego de rede, o NIDSs baseado em DNN deve passar por atualizações periódicas, um procedimento que frequentemente leva várias semanas ou até meses para ser concluído. Enquanto isso, o modelo implantado no ambiente de produção, embora desatualizado, deve manter a confiabilidade de sua classificação.

Para enfrentar tal desafio, implementamos o procedimento de classificação através de uma classificação com a opção de rejeição para aumentar o tempo de vida do modelo. O modelo proposto adere ao fluxo convencional de NIDS baseados em DNN implementados utilizando saídas antecipadas (Figura 1). Dado um modelo DNN com N ramificações, onde a ramificação final produz um valor de confiança de classificação $\alpha = \{\alpha_{normal}, \alpha_{intrusion}\}$ para cada evento de entrada x. Aqui, α_{normal} denota a confiança do evento para a classe *normal*, e $\alpha_{intrusion}$ a confiança do evento para *ataque*, de forma que $\alpha \in \mathbb{R}[0, 1]$. O *Módulo de rejeição* aceita ou rejeita a classificação baseado nos valores de confiança α $vs.$ um limiar de rejeição previamente definido, t_{rej} (vide Figura 1). Para atingir tal objetivo, a implementação do módulo utiliza a função de rejeição, determinada pelo algoritmo 1.

onde $\emptyset$ denota eventos nos quais a última ramificação do modelo DNN provavelmente classificou incorretamente. Como resultado, o *Módulo de Rejeição* suprime classificações não confiáveis, medidas por seus valores de confiança associados à última ramificação da DNN. É importante notar que o limiar de rejeição deve ser determinado de acordo

Algorithm 1 Rejector Module (α, t_{rej})

```
for Event do
    if α ≤ t_rej then
        return ∅
    else
        return α
    end if
end for
```

com as necessidades do operador de rede. Um limiar mais alto de rejeição aumentará a confiabilidade do sistema, mas resultá em um maior número de eventos rejeitados. Por outro lado, um limiar mais baixo aceitará mais eventos, mas deixará o sistema exposto à classificações potencialmente não confiáveis.

4.2. Otimização Multi-objetivo

Detecção de intrusão baseado em rede para dispositivos com recursos limitados deve ser realizada com os mínimos requisitos de processamento enquanto mantém os níveis de acurácia e confiabilidade. Para tratar esse desafio, nós construímos o modelo com uma tarefa de otimização multi-objetivo.

Consideramos um modelo DNN h implementado com N ramificações, acoplado com um *Módulo de rejeição* na ramificação final (vide Seção 4.1). Nesse caso, o objetivo da otimização multi-objetivo é encontrar $N-1$ limiares associados às ramificações t_b, juntamente com o limiar do *Módulo de rejeição* t_{rej}. Portanto, consideramos dois objetivos baseados nos custos de processamento do sistema e acurácia. Assumimos que o primeiro objetivo é um resultado direto de aceitar um número maior de eventos nas ramificações iniciais, enquanto o segundo objetivo é relacionado ao aumento da taxa de rejeição, que, por sua vez, aumenta custos de processamento devido a mais eventos chegarem na última ramificação. Dessa forma, podemos trabalhar o multi-objetivo resolvendo a seguinte equação:

$$\begin{aligned} &\underset{t_{rej},\{t_b^1,\ldots,t_b^{N-1}\}}{\arg\min} \; tempo(h(\mathcal{D}, t_{rej}, \{t_b^1, ..., t_b^{N-1}\})) \\ \text{e} \quad & \\ &\underset{t_{rej},\{t_b^1,\ldots,t_b^{N-1}\}}{\arg\min} \; erro(h(\mathcal{D}, t_{rej}, \{t_b^1, ..., t_b^{N-1}\})) \end{aligned} \tag{2}$$

onde h denota as múltiplas ramificações do modelo DNN acoplado com nosso *Módulo de Rejeição* (vide Figura 1). Aqui, $tempo$ mede o tempo de inferência do modelo h em um dado conjunto de dados $\mathcal{D}$ quando usando uma taxa de rejeição t_{rej} e um conjunto de limiares de ramificação t_b. Por sua vez, $erro$ mede a taxa de erro com o mesmo conjunto de limiares. Como resultado, nosso esquema proposto tem como objetivo identificar os limiares ótimos para o sistema, que melhoram o tempo de inferência ao mesmo tempo que minimizam as taxas de erro.

Tabela 1. Taxa média de detecção de eventos (eventos/seg).

DNN	Ambiente	
	Desktop	
	CPU	GPU
AlexNet	247.34	17,609

5. Resultados

Nesta seção, aprofundamos o desempenho das técnicas de DNN em relação em relação à acurácia e aos requisitos de processamento quando confrontados com mudanças no comportamento do tráfego de rede. Mais especificamente, apresentamos inicialmente o conjunto de dados utilizado nos experimentos conduzidos nesse trabalho, seguido de uma avaliação do desempenho das técnicas de DNN sobre esse conjunto de dados.

5.1. MAWIFlow

Para garantir uma avaliação realista, utilizamos o conjunto de dados *MAWIFlow*, um conjunto de dados de intrusão disponível publicamente, contendo tráfego de rede real, válido e rotulado extraído de ambientes de produção durante um longo período. Para alcançar essas características, o conjunto de dados é construído a partir dos arquivos de tráfego do grupo de trabalho MAWI [MAWI 2023]. Ele inclui tráfego de rede do *Samplepoint-F* do MAWI, um link de trânsito de rede entre o Japão e os EUA, coletado diariamente em intervalos de 15 segundos. Os dados de rede são coletados diariamente, resultando em um arquivo de rede PCAP para cada dia do período avaliado, totalizado mais de 7TB de dados e abrangendo mais de 70 bilhões de fluxos de rede. Para este trabalho, os dados de rede coletados no ano de 2016 foram utilizados. Os dados coletados são organizados em fluxos de rede baseados nos nós e serviços envolvidos em cada comunicação. Cada fluxo de rede representa um segmento de 15 segundos de dados de cliente/serviço e servidor/serviço, que é subsequentemente resumido em um conjunto de características associado. Para o propósito deste trabalho, extraímos 58 características do trabalho de Moore [Moore 2005]. Para atribuir rótulos, nosso trabalho utiliza os algoritmos não-supervisionados de aprendizagem de máquina MAWILab [Fontugne et al. 2010] que identificam anomalias de rede que são então rotuladas como ataques em nosso conjunto de dados.

5.2. Perseguindo um alvo em movimento

A primeira avaliação tem como objetivo responder duas Questões de Pesquisas (QPs):

- (***QP1***) *Qual o desempenho em detecção de intrusão de técnicas de DNN amplamente utilizadas?*
- (***QP2***) *Quais são os custos computacionais das técnicas avaliadas?*

Nós avaliamos o desempenho de um arquiteturas de DNN amplamente utilizada, denominada AlexNet. A arquitetura foi ajustada para se adequar ao formato tabular do *MAWIFlow*. Mais precisamente, nós transformamos a entrada em um canal único, em uma matriz de dimensão 8x8. Foi então expandido utilizando uma camada totalmente conectada com 2304 neurônios e uma função de ativação *relu*. A camada de saída é então ajustada para 48x48x1 utilizando agrupamento médio (*average pooling*), servindo então como entrada para a arquitetura DNN. A DNNs foi então treinadas utilizando o otimizador *adam*, executando o treinamento por 1.000 épocas. Nós utilizamos *categorical*

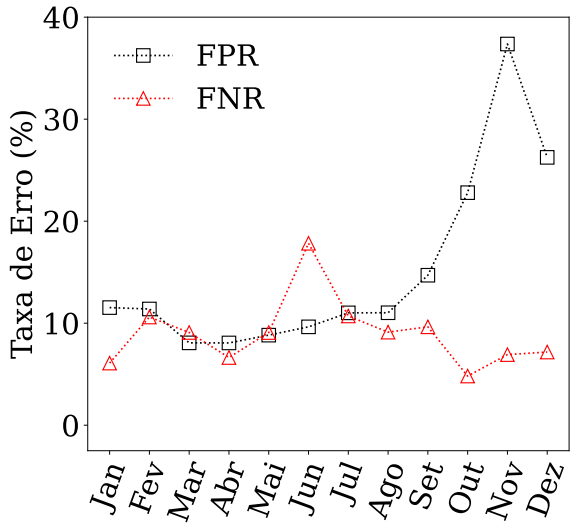

Figura 2. Tendência da acurácia em um ano do classificador utilizado, sem atualizações periódicas do modelo. Classificador for treinados em janeiro e avaliados nos meses subsequentes, sem atualizações

cross-entropy como a função de perda. A taxa de aprendizado foi de $0,001$ com um avaliador que interrompe o treinamento se não houver melhoria na acurácia de validação após 50 épocas. Esse modelo foi implementados utilizando a API PyTorch versão $2.1.0$. O classificador é avaliado em termos de taxas de Falso Positivo (FP) e Falso Negativo (FN). FP refere-se à taxa de eventos normais que são incorretamente classificados como ataques, enquanto FN apresenta a taxa de ataques que foram incorretamente classificadas como tráfego normal.

O experimento inicial foi projetado para responder a *QP1* e avalia o desempenho da classificação da técnica de detecção de intrusão escolhida no conjunto de dados *MAWIFlow* quando encontra as mudanças de comportamento do tráfego de rede no tempo. Para atingir esse objetivo, nós treinamos a arquitetura DNN selecionada usando os dados de janeiro do *MAWIFlow*. Nós avaliamos o desempenho do modelo nos demais meses do ano sem nenhuma atualização periódica no modelo. A Figura 2 mostra a acurácia mensal da arquitetura DNN escolhida, onde é notável uma redução significativa da acurácia no tempo. Por exemplo, o modelo, como visto na Figura 2, sofre um aumento na sua taxa FP de até 25% quando comparado com período de treinamento (Jan. vs Nov.). As técnicas de detecção de intrusão avaliadas têm dificuldade em lidar com a evolução dos padrões do tráfego de rede no tempo.

O segundo experimento buscou responder a *QP2* e examina os custos computacionais associados às técnicas selecionadas. Nós avaliamos os custos computacionais médios de inferência em um ambientes Desktop, equipado com um 16 núcleos de processamento Intel Xeon E5-2640 v3, 32 GB de memória e uma GPU Nvidia Tesla T4, rodando Ubuntu Linux 22.04. A Tabela 1 mostra a taxa de inferência média para a DNN avaliadas na plataformas selecionada. Nesse cenário, uma taxa média de ≈ 247.34 eventos por segundo se mostra insuficiente para lidar com milhares de eventos de rede que um dispositivo pode encontrar quando instalado em uma rede real. Portanto, além de lidar com as nuanças no comportamento do tráfego de rede no tempo, os esquemas propostos também devem ser capazes de realizar essa tarefa com demandas mínimas de recursos de processamento.

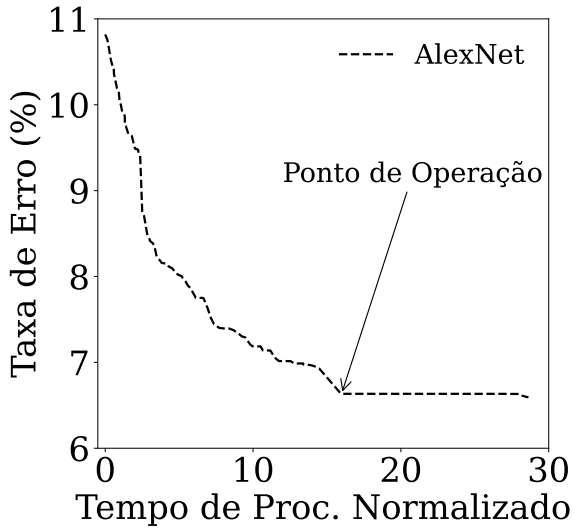

Figura 3. Otimização Multi-objetivo para a arquitetura avaliada. Taxa de erro foi medida em *MAWIFlow* meses de Fev. e Mar.

5.3. Avaliações

Nós avaliamos nosso sistema respondendo as seguintes QPs:

- (***QP3***) *Como nossa otimização multi-objetivo proposta melhora a confiabilidade do sistema?*
- (***QP4***) *Como nossa técnica de rejeição melhora o desempenho da classificação?*
- (***QP5***) *Quais são os custos computacionais do esquema proposto?*

As subseções a seguir apresentam detalhes adicionais sobre a implementação do nosso modelo e seu desempenho.

5.4. Construção do modelo

Nós implementamos nosso modelo proposto utilizando a arquitetura de DNN AlexNet que será avaliada posteriormente. Dados que usamos saídas antecipadas, adicionamos à DNN uma ramificação intermediária. Adicionamos uma saída antecipada entre as 1^{st} e 2^{nd} camadas convolucionais. A saída antecipada da rede AlexNet planifica a camada de saída anterior e aplica uma camada totalmente conectada com 1600 neurônios de entrada e 2 neurônios de saída. Portanto, arquitetura de DNN avaliada é composta por duas ramificações, abrangendo as camadas adicionadas da primeira ramificação, enquanto a última ramificação compreende a saída tradicional da DNN. O modelos modificado é treinado utilizando a função de perda *categorical cross-entropy* para cada ramificação (vide Eq. 1) com o peso w de cada camada definido como 1.0. A taxa de aprendizado foi definida em 0.001. Esse modelo foi implementados utilizando a API PyTorch versão $2.1.0$.

5.5. Inferência Rápida e Detecção de Intrusão Confiável

Nosso primeiro experimento tem como objetivo responder a *QP3* e avaliar como nossa otimização multi-objetivo proposta pode diminuir o custo computacional da inferência enquanto mantém a acurácia do sistema. Nós implementamos nosso esquema utilizando *Non-dominated Sorting Genetic Algorithm II (NSGA-II)*) [Deb et al. 2002] utilizando a

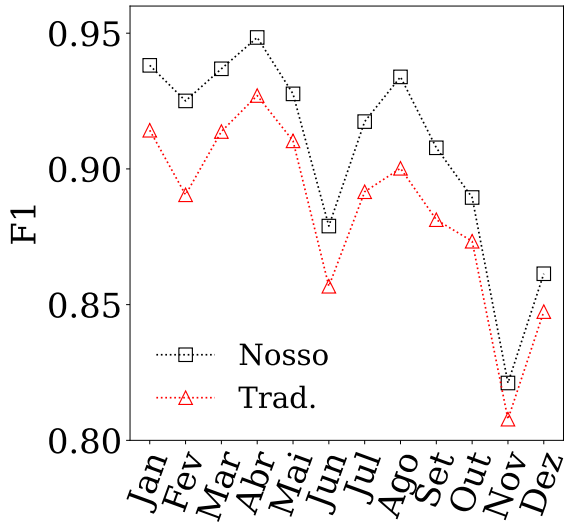

Figura 4. Comparação de acurácia e tempo de processamento da AlexNet com e sem nosso esquema no conjunto de dados MAWIFlow.

API *pymoo*. O NSGA-II usa uma população de tamanho 100, 1000 gerações, um *crossover* de 0.9 e uma probabilidade de mutação de 1.0. A seleção de características multi-objetivo procura resolver a Equação 2 enquanto otimiza o limiar utilizado para a primeira ramificação (t_b^1) e o limiar do *Módulo de Rejeição* (t_{rej}), considerando dois objetivos, denominados *tempo* e *erro*. Nós medimos *tempo* normalizando o tempo de processamento medido no conjunto de dados de teste entre o tempo de processamento mínimo (primeira ramificação) e o tempo máximo (última ramificação), como determinado pela arquitetura DNN. De forma similar, nós medimos *erro* como a média entre as taxas FP e FN. Nós também consideramos uma restrição de otimização que a taxa mínima de aceitação na última ramificação seja de 90% (Figura 1, *Módulo de Rejeição*)

A Figura 3 mostra a curva de Pareto para a técnica de otimização multi-objetivo proposta, onde nota-se uma relação de escolha entre a acurácia do modelo $vs.$ a taxa de detecção. Por exemplo, a otimização multi-objetivo permite que o operador de rede alcance uma taxa de erro de apenas 6.6% enquanto requer apenas $\approx 16\%$ dos custos computacionais (Figura 3, *Ponto de Operação* para AlexNet). Aprofundamos a investigação nos ganhos do nosso esquema quando comparados à abordagem tradicional. Nossa otimização multi-objetivo pode reduzir a taxa de erro do sistema em 3.3 ao mesmo tempo que reduz o tempo médio de processamento em até 82%, quando utilizando os limiares de ramificação e de rejeição definidos no Ponto de Operação destacado na Figura 3..

Para responder a *QP4*, nós analisamos o desempenho do nosso esquema sobre o conjunto de dados *MAWIFlow*, com a adição do *Módulo de rejeição* na última camada. Nesse caso, nós definimos o limiar da primeira ramificação (t_b^1) e o limiar do *Módulo de rejeição* (t_{rej}) para os valores do Ponto de Operação destacado na Figura 3. A DNNs é então avaliada utilizando o conjunto de dados *MAWIFlow* sem atualizações de modelo. A Figura 5 mostra a acurácia do nosso esquema sem atualizações periódicas do modelo com o *Módulo de rejeição*. Nosso esquema proposto apresenta melhores taxas de acurácia conforme o tempo passa. A Figura 4 aprofunda a investigação dos benefícios na acurácia do nosso modelo com AlexNet quando comparado à abordagem tradicional. Nosso esquema consegue melhorar o F1 por uma média de 0.03, com um ganho de até 0.06 em

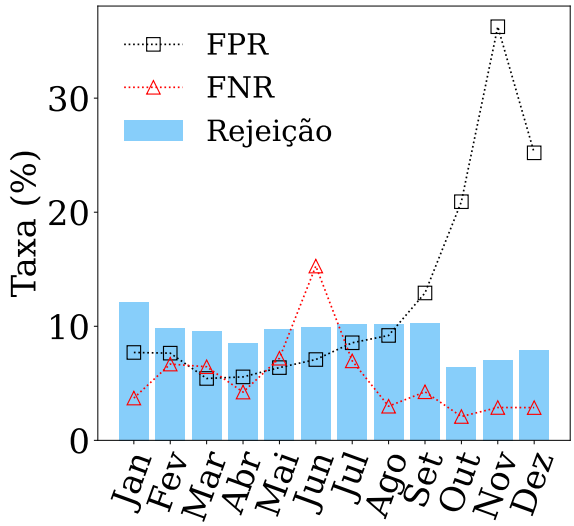

Figura 5. Acurácia em um ano no esquema proposto. O classificador é treinado em janeiro e avaliado nos demais meses, sem atualização.

agosto para a DNN AlexNet.

Para responder a *QP5* nós investigamos os benefícios em processamento do nosso esquema. Para isso, medimos o custo mensal médio de processamento de eventos, lembrando que ele varia de acordo com a proporção de eventos que são aceitos na primeira ramificação. Em média, nosso modelo requer apenas $\approx 25\%$ de processamento quando comparado à abordagem tradicional. Na prática, nosso modelo pode reduzir a taxa de erro, ao mesmo tempo em que pode aprimorar significativamente os tempos médios de processamento, abrindo perspectivas para a implementação de NIDSs utilizando DNN em dispositivos com recursos limitados.

6. Conclusão

As abordagens de NIDS enfrentam dificuldades para realizar detecções confiáveis em dispositivos IoT. Elas não apenas requerem níveis impraticáveis de recursos de processamento, mas também enfrentam desafios ao lidar com mudanças de comportamento no tráfego de rede. Esse artigo introduziu um novo esquema de detecção de intrusão incorporando saídas antecipadas e um classificador com uma opção de rejeição. O primeiro elemento é projetado para aumentar a taxa de detecção de intrusão em dispositivos com recursos limitados, enquanto o outro tem como objetivo garantir a confiabilidade da classificação mesmo na presença de novos comportamentos de tráfego de rede. Os experimentos mostram que nosso esquema proposto reduz custos de processamento ao mesmo tempo em que aumentam a acurácia da classificação, graças à abordagem de otimização multi-objetivo. Como parte de trabalhos futuros, planejamos incorporar atualizações de modelo em eventos rejeitados.

Agradecimentos

Este trabalho foi parcialmente financiado pelo Conselho Nacional de Desenvolvimento Científico e Tecnológico (CNPq), termos 304990/2021-3 e 407879/2023-4, e 302937/2023-4

Referências

Abreu, V., Santin, A. O., Viegas, E. K., and Stihler, M. (2017). A multi-domain role activation model. In *2017 IEEE International Conference on Communications (ICC)*. IEEE.

Al-Garadi, M. A., Mohamed, A., Al-Ali, A. K., Du, X., Ali, I., and Guizani, M. (2020). A survey of machine and deep learning methods for internet of things (IoT) security. *IEEE Communications Surveys & Tutorials*, 22(3):1646–1685.

Deb, K., Pratap, A., Agarwal, S., and Meyarivan, T. (2002). A fast and elitist multiobjective genetic algorithm: NSGA-II. *IEEE Transactions on Evolutionary Computation*, 6(2):182–197.

dos Santos, R. R., Viegas, E. K., Santin, A. O., and Tedeschi, P. (2023). Federated learning for reliable model updates in network-based intrusion detection. *Computers amp; Security*, 133:103413.

Fontugne, R., Borgnat, P., Abry, P., and Fukuda, K. (2010). 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)*.

Ge, M., Syed, N. F., Fu, X., Baig, Z., and Robles-Kelly, A. (2021). Towards a deep learning-driven intrusion detection approach for internet of things. *Computer Networks*, 186:107784.

Geremias, J., Viegas, E. K., Santin, A. O., Britto, A., and Horchulhack, P. (2022). Towards multi-view android malware detection through image-based deep learning. In *2022 International Wireless Communications and Mobile Computing (IWCMC)*. IEEE.

Horchulhack, P., Viegas, E. K., Santin, A. O., Ramos, F. V., and Tedeschi, P. (2024a). Detection of quality of service degradation on multi-tenant containerized services. *Journal of Network and Computer Applications*, 224:103839.

Horchulhack, P., Viegas, E. K., Santin, A. O., and Simioni, J. A. (2024b). Network-based intrusion detection through image-based cnn and transfer learning. In *2024 International Wireless Communications and Mobile Computing (IWCMC)*. IEEE.

Laskaridis, S., Kouris, A., and Lane, N. D. (2021). Adaptive inference through early-exit networks. In *Proceedings of the 5th International Workshop on Embedded and Mobile Deep Learning*. ACM.

Li, E., Zeng, L., Zhou, Z., and Chen, X. (2020). Edge AI: On-demand accelerating deep neural network inference via edge computing. *IEEE Transactions on Wireless Communications*, 19(1):447–457.

Ma, L., Chai, Y., Cui, L., Ma, D., Fu, Y., and Xiao, A. (2020). A deep learning-based DDoS detection framework for internet of things. In *IEEE International Conference on Communications (ICC)*. IEEE.

MAWI (2023). MAWI Working Group Traffic Archive - Samplepoint F.

Molina-Coronado, B., Mori, U., Mendiburu, A., and Miguel-Alonso, J. (2020). Survey of network intrusion detection methods from the perspective of the knowledge disco-

very in databases process. *IEEE Transactions on Network and Service Management*, 17:2451–2479.

Moore, A. (2005). Discriminators for use in flow-based classification. In *Dept. Comput. Sci., Univ. London, London, U.K., Rep. RR-05-13*.

Santos, R. R. d., Viegas, E. K., Santin, A. O., and Cogo, V. V. (2023). Reinforcement learning for intrusion detection: More model longness and fewer updates. *IEEE Transactions on Network and Service Management*, 20(2):2040–2055.

Seifeddine, W., Adjih, C., and Achir, N. (2021). Dynamic hierarchical neural network offloading in IoT edge networks. In *2021 10th IFIP International Conference on Performance Evaluation and Modeling in Wireless and Wired Networks (PEMWN)*. IEEE.

SonicWall (August, 2023 (accessed October 5, 2023)). *Mid-Year Update: 2023 SonicWall Cyber Threat Report*.

Viegas, E., Santin, A., Neves, N., Bessani, A., and Abreu, V. (2017). A resilient stream learning intrusion detection mechanism for real-time analysis of network traffic. In *GLOBECOM 2017 - 2017 IEEE Global Communications Conference*. IEEE.

Wang, Y., Qin, G., Zou, M., Liang, Y., Wang, G., Wang, K., Feng, Y., and Zhang, Z. (2023). A lightweight intrusion detection system for internet of vehicles based on transfer learning and MobileNetV2 with hyper-parameter optimization. *Multimedia Tools and Applications*.

Zhang, J., Li, F., and Ye, F. (2020). An ensemble-based network intrusion detection scheme with bayesian deep learning. In *IEEE International Conference on Communications (ICC)*. IEEE.