UNIVERSIDADE ESTADUAL PAULISTA JÚLIO DE MESQUITA FILHO

FACULDADE DE CIÊNCIAS DE BAURU

CURSO DE BACHARELADO SISTEMAS DE INFORMAÇÃO

Trabalho de conclusão de curso

Darryê Roberto da Silva Mellin

DESENVOLVIMENTO DE UM SISTEMA PARA DETECÇÃO DE TRÁFEGO DE REDE MALICIOSO

Bauru
2025

UNIVERSIDADE ESTADUAL PAULISTA JÚLIO DE MESQUITA FILHO

FACULDADE DE CIÊNCIAS DE BAURU

CURSO DE BACHARELADO SISTEMAS DE INFORMAÇÃO

Trabalho de Conclusão de Curso

Darryê Roberto da Silva Mellin

DESENVOLVIMENTO DE UM SISTEMA PARA DETECÇÃO DE TRÁFEGO DE REDE MALICIOSO

Trabalho apresentado como exigência para a conclusão do Curso de Bacharelado em Sistemas de Informação da Faculdade de Ciências – UNESP Campus Bauru.

Orientador: Prof. Dr. Kelton Augusto Pontara da Costa

Bauru
2025

M526d

Mellin, Darryê Roberto da Silva
Desenvolvimento de um sistema para detecção de tráfego de rede malicioso / Darryê Roberto da Silva Mellin. -- Bauru, 2025
29 f. : il., tabs.

Trabalho de conclusão de curso (Bacharelado - Sistemas de Informação) - Universidade Estadual Paulista (UNESP), Faculdade de Ciências, Bauru
Orientador: Kelton Augusto Pontara da Costa

1. Malware (Software de computador). 2. Inteligência artificial. 3. Sistemas de detecção de intrusão (Medidas de segurança). I. Título.

Sistema de geração automática de fichas catalográficas da Unesp. Dados fornecidos pelo autor(a).

Darryê Roberto da Silva Mellin

DESENVOLVIMENTO DE UM SISTEMA PARA DETECÇÃO DE TRÁFEGO DE REDE MALICIOSO

Trabalho apresentado como exigência para a conclusão do Curso de Bacharelado em Sistemas de Informação da Faculdade de Ciências – UNESP Campus Bauru.

Banca Examinadora

Prof. Dr. Kelton Augusto Pontara da Costa
Orientador
Universidade Estadual Paulista "Júlio de Mesquita Filho"
Faculdade de Ciências
Departamento de Computação

Prof. Dr. Clayton Reginaldo Pereira
Universidade Estadual Paulista "Júlio de Mesquita Filho"
Faculdade de Ciências
Departamento de Computação

Prof. Dr. José Remo Ferreira Brega
Universidade Estadual Paulista "Júlio de Mesquita Filho"
Faculdade de Ciências
Departamento de Computação

RESUMO

O trabalho apresenta um software baseado em Npcap com inteligência artificial que é capaz de analisar tráfego de rede em busca de IPs que apresentam comportamento anormal ou suspeito, em outras palavras, que esteja transmitindo dados incomuns via rede, podendo assim, indicar a existência de um software malicioso, ou um ataque. A forma de detecção escolhida baseia-se no uso de inteligência artificial para reconhecer os padrões de tráfego na rede no intuito de identificar possíveis irregularidades. Em outras palavras, a forma de detecção foi baseada em anomalias. O software, com detecção baseada em anomalias, apresenta desempenho satisfatório e promissor demonstrando bom equilíbrio entre detecções e falsos positivos.

Palavras-chave: Malware, Detecção, Inteligência artificial

ABSTRACT

This work presents an Npcap-based software with artificial intelligence capable of analyzing network traffic in search of IPs exhibiting abnormal or suspicious behavior, in other words, those transmitting unusual data over the network, which may indicate the existence of malicious software or an attack. The chosen detection method is based on the use of artificial intelligence to recognize network traffic patterns in order to identify possible irregularities. In other words, the detection method was anomaly-based. The software, with anomaly-based detection, shows satisfactory and promising performance, demonstrating a good balance between detections and false positives.

Keywords: Malware, Detection, Artificial intelligence

LISTA DE FIGURAS

LISTA DE TABELAS

SUMÁRIO

1 INTRODUÇÃO

Tráfego de rede malicioso pode ser originado de diversas fontes, como computadores infectados com malware, dispositivos de IoT vulneráveis, uma invasão da rede, ataques de negação de serviço entre várias outras formas de ataque. Tanto a rede interna quanto a externa da organização podem ser fontes de vulnerabilidades, com isso em mente, os profissionais da segurança de informação tem o trabalho árduo de impedir uma intrusão na rede, mas, isso nem sempre é possível, no caso da rede ser comprometida, é ideal que o ator mal intencionado seja identificado e neutralizado o mais rápido possível, a fim de diminuir os danos causados.

Atualmente a existência de softwares maliciosos que tem como objetivo obter dados empresariais ou pessoais é um problema crescente, como foi revelado no comunicado à imprensa da Kaspersky em janeiro de 2024. Não só o número de malwares detectados aumentou em 2023 em relação a 2022, como também, das formas de ataque. O uso de *backdoors*, que é um meio de acesso a um sistema computacional que ignora os processos típicos de autenticação, em redes corporativas, além de malwares disfarçados como arquivos do Office, também aumentaram em 53% em relação a anos anteriores. (KASPERSKY, 2024)

Outra forma de ataque que pode gerar grandes prejuízos para a organização são os ataques de negação de serviço. O grande avanço na área de computação na nuvem criou um novo alvo para ataques DDoS ou DoS, ataques aos serviços de nuvem de uma empresa podem paralisar completamente o seu funcionamento, o que significa um prejuízo crescente enquanto o problema não é resolvido. Um relatório da Kaspersky indica um aumento na demanda e disponibilidade de *botnets* para ataques DDoS em 2023. (KASPERSKY, 2023)

Sendo assim, o objetivo principal do trabalho é criar um software baseado em inteligência artificial, que fique em execução e analise o tráfego de rede com o objetivo de encontrar anomalias, ou seja, que trabalhe como “vigia” e detecte quando um ip está transmitindo um fluxo de dados suspeito. Dessa forma, será possível identificar uma intrusão na rede e a sua respectiva origem.

Essa forma de detecção é conhecida como *Anomaly-Based* detection (detecção baseada em anomalia), que consiste em aprender os padrões de tráfego na rede para poder identificar os desvios, a grande vantagem desse tipo de detecção de malwares é que não é necessário ter um banco de assinaturas de malwares conhecidos, ou seja, ela é viável quando se trata de identificar malwares novos e desconhecidos. Uma desvantagem desse tipo de detecção é que ela pode resultar em um número considerável de falsos positivos, dada a natureza volátil do fluxo de dados na rede. ZHAO et al. (2015)

Outra forma de detecção de malwares é chamada de *Signature-Based* detection (detecção baseada em assinatura), diferente da detecção baseada em anomalia, essa técnica consiste no uso de uma base de dados de malwares já conhecidos e suas assinaturas, a principal vantagem dessa forma de detecção é que ela dificilmente gera falsos positivos, mas, torna-se ineficiente quando tratamos da detecção de novos malwares, que não estão presentes na base de dados ou, malwares de código polimórfico. ZHAO et al. (2015)

1.1 Detalhamento do problema

No quadro geral o número de malwares novos tem aumentado a cada ano, segundo um relatório da Kaspersky em 2022, o número de novos malwares detectados diariamente aumentou em 5%, passando de 380.000 detecções diárias em 2021 para 400.000 em 2022. São números que estão em ascensão quando comparados a quaisquer anos anteriores, ou seja, cada ano tem um número maior de detecções em relação ao anterior.(TEAM KASPERSKY, 2021)

Esse é apenas um exemplo de anos anteriores, como citado anteriormente, no relatório de 2023, os números são 3% superiores em relação a 2022, passando para 411000 detecções diárias. Outro grande problema, é o aumento no número de malwares disfarçados de arquivos do Office e PDFs.

Segundo Fabio Assolini, diretor da Equipe Global de Pesquisa e Análise da Kaspersky para a América Latina:

> “Novos grupos surgem devido à proliferação da IA, que está sendo usada na criação de mensagens de phishing mais convincentes. Nestes tempos, é essencial, tanto para as grandes organizações como para cada pessoa, adotar soluções de segurança confiáveis.” (Kaspersky, 2024).

O aumento no número de ataques de *phishing* significa que malwares são introduzidos com facilidade no sistema computacional da empresa, e desde que não sejam detectados podem ficar meses coletando informações.

1.2 Soluções semelhantes

Hoje existem inúmeras ferramentas usadas na proteção de dados e ativos de software, sejam eles empresariais ou pessoais. Essas ferramentas situam-se na mesma área que este trabalho deseja atingir e podem ser consideradas soluções próximas.

1.2.1 Softwares

Primeiramente temos o uso dos antivírus convencionais, como Kaspersky (2025), Norton (2025) e Bitdefender (2025). Esses softwares são voltados para proteção geral contra malwares, utilizando de bancos de dados para realizar o scan por malwares por assinatura.

Os softwares de antivírus são principalmente voltados para os usuários que não possuem o conhecimento técnico de profissionais de segurança da informação, e podem ser usados em conjunto com ferramentas especializadas.

Existem também ferramentas voltadas para profissionais de segurança da informação como o *sniffer* de rede Snort (2025) e o software de captura de pacotes Wireshark (2025). Ambos são utilizados para detecção de intrusão na rede, mas, o uso desses softwares requer um profundo conhecimento técnico da área de redes de computadores e segurança da informação.

1.2.2 Pesquisas

O dataset do MIT (*Massachusetts Institute of Technology*), de 1999, intitulado “**1999 DARPA Intrusion Detection Evaluation Dataset**” trata da criação de um sistema de detecção de intrusão, baseado em anomalias, para tentar detectar tráfego malicioso ocorrendo em meio ao tráfego comum da rede. O dataset foi construído de maneira aditiva com o dataset de 1998.

Os dados de treinamento foram divididos em três subconjuntos, cada um deles contendo uma semana de tráfego capturado de uma rede simulada. Tanto a primeira, quanto a terceira semana de dados tratavam de tráfego livre de qualquer ataque, a segunda continha tráfego misto (LINCOLN LABORATORY, 1999)

2 FUNDAMENTAÇÃO TEÓRICA

Este capítulo tem como objetivo definir a base conceitual essencial desse trabalho.

2.1 Malware

Malwares podem ser classificados em três grandes categorias, vírus, trojan e worm. Os softwares maliciosos tem uma grande variedade, muitas vezes, sendo feitos especificamente para um ataque que explora uma vulnerabilidade específica, então para diferenciá-los podemos tomar como diferenciadores as suas características mais básicas, como por exemplo a necessidade de um usuário executar o arquivo. (STALLINGS, 2008)

Um vírus tem que ser executado por um usuário, e geralmente possuem a capacidade de se espalhar pelo sistema modificando outros programas. Um worm não precisa de um usuário para ser executado e se espalha sozinho pela rede. Um trojan normalmente consiste de uma ferramenta que possui funções ocultas.

2.2 Vulnerabilidade

Uma vulnerabilidade no âmbito da computação é basicamente qualquer tipo de brecha ou ponto de falha em um sistema computacional, que pode ser explorada por um atacante. Vulnerabilidades podem estar presentes em todo o ambiente de uma organização, como por exemplo, a falta de treinamento ou instrução adequada dos funcionários, regras mal elaboradas de firewall e design falho da rede interna.(Hintzbergen, Hintzbergen, Smulders, Baars, 2018).

2.3 Tipos de ataque

Para delimitar com maior clareza o escopo do trabalho, os ataques que o software será treinado para reconhecer serão detalhados. Esses são os ataques contidos na base de dados utilizada para o treinamento do modelo de inteligência artificial:

a) **DoS**: É um ataque de negação de serviço. Dentro deste tipo de ataque existem diversas subdivisões, denominadas a partir do nome das ferramentas utilizadas para os ataques, elas são: *Hulk, GoldenEye, Slowloris* e *slowhttptest;*
b) ***PortScan***: É uma técnica utilizada para descobrir quais portas de um sistema estão abertas, podendo assim descobrir quais serviços estão disponíveis (HTTP, SSH ou FTP por exemplo);
c) **DDoS**: É um ataque de negação de serviço distribuído, o ataque faz uso de botnets para criar as requisições, o que o torna mais difícil de ser detectado;
d) **FTP-Patator**: É um ataque de força bruta, ou seja, o atacante realiza combinações de usuário e senha até encontrar uma válida, no caso, o protocolo FTP (*File Transfer Protocol*) é o alvo;
e) **SSH-Patator**: também é um ataque de força bruta, mas tem como alvo o protocolo SSH (*Secure Shell*);
f) **Bot**: Trata-se de um dispositivo infectado que pode ser controlado remotamente, geralmente são utilizados para realizar tarefas automatizadas, como ataques DDoS, spam de e-mails ou coleta de dados. Quando esse bot faz parte de uma rede de bots essa rede é chamada de botnet;
g) ***Web Attack Bruteforce***: É um ataque de força bruta que tem como alvo aplicações web, como páginas de login;
h) ***Web Attack XSS***: É um ataque que tem como objetivo injetar código malicioso em sites legítimos com o intuito de roubar cookies, tokens de acesso e outras informações armazenadas pelo navegador;

i) ***Infiltration***: Refere-se a obter acesso a rede interna, a partir do acesso o atacante pode extrair informações, introduzir malwares, atrapalhar operações ou destruir informações;
j) ***Web Attack SQL injection***: O atacante injeta comandos SQL em formulários de aplicações web, podendo roubar, modificar ou destruir o banco de dados;
k) ***Heartbleed***: É uma vulnerabilidade da biblioteca de criptografia OpenSSL, ela permite que o atacante leia áreas da memória do servidor que não deveriam ser acessíveis, o que pode expor informações importantes, como senhas e chaves de acesso.

3 DESCRIÇÃO DO SOFTWARE

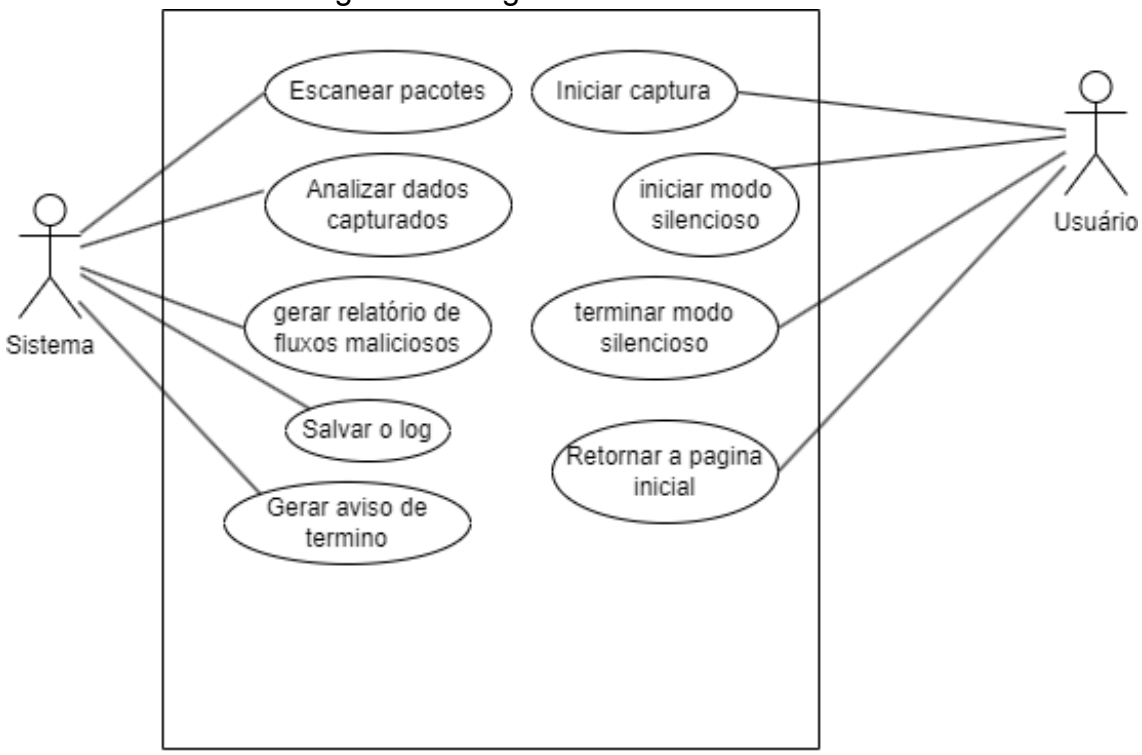
Primeiramente, o software monolítico foi desenvolvido em *threads* que são interligadas por uma *thread* de controle geral. A *thread* de processamento de pacotes foi desenvolvida utilizando a biblioteca de captura de pacotes para o Windows Npcap (2025), além de um script Python para conversão dos pacotes capturados em um fluxo de dados. A *thread* de inteligência artificial foi desenvolvida utilizando a linguagem Python e a linguagem C, essa thread é responsável por criar os arquivos de log e da captura mais recente. Por fim, a *thread* de controle geral foi desenvolvida na linguagem C, com a interface gráfica utilizando a biblioteca WinApi disponibilizada pela Microsoft (2025).

Com isso, o software foi desenvolvido primariamente em C e Python.

3.1 Diagrama de Casos de Uso

O diagrama de casos de uso tem como objetivo detalhar as funções do software e quais delas serão utilizadas pelo próprio sistema ou pelo usuário. (Figura 1)

Figura 1 - Diagrama de casos de uso

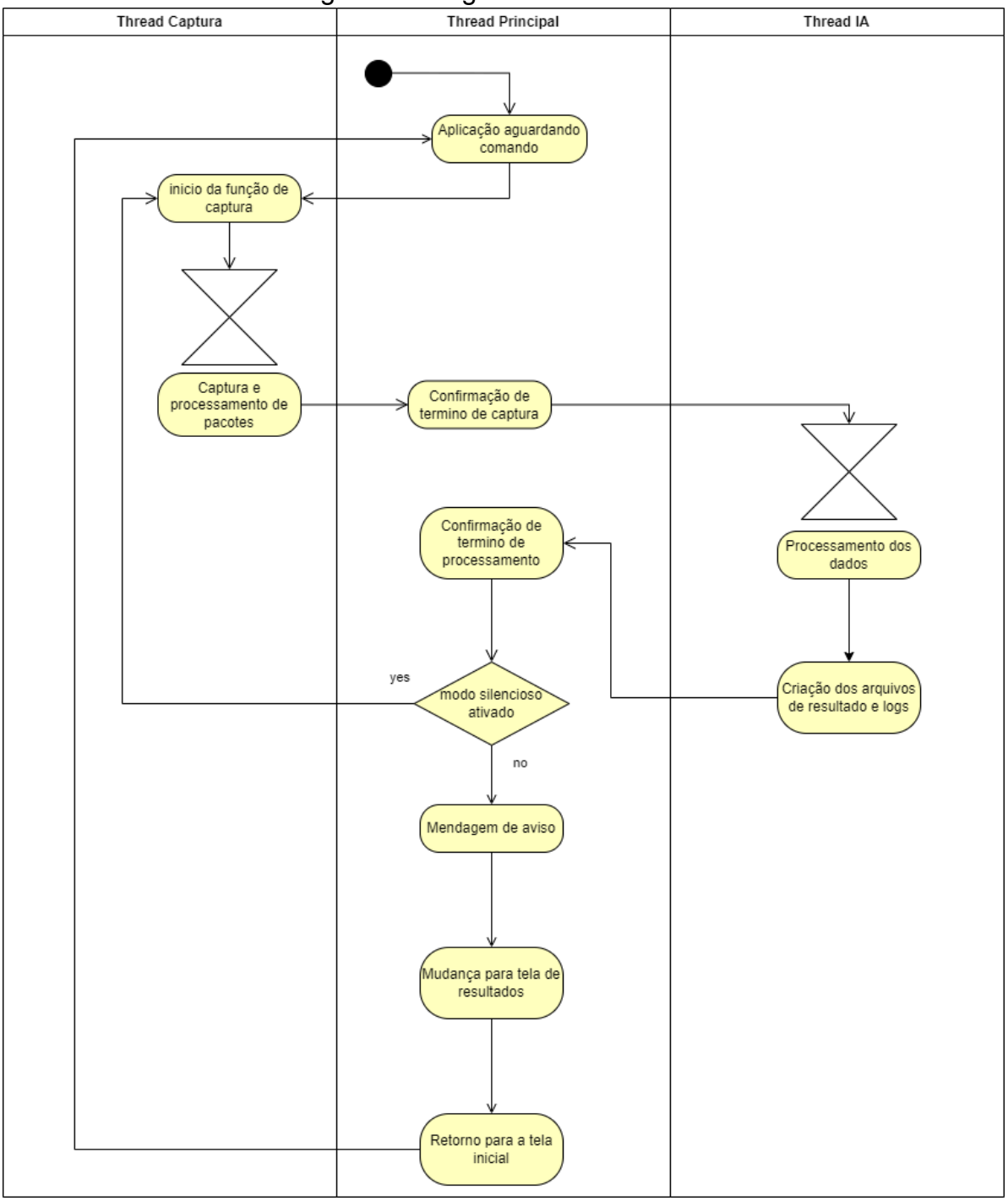


Fonte: Elaborado pelo autor.

3.2 Diagrama de Atividades

O diagrama de atividade tem como objetivo ilustrar como as atividades do sistema interagem entre si durante sua execução. A Figura 2, apresenta o fluxo completo de atividade do sistema (Figura 2).

Figura 2 - Diagrama de Atividades



Fonte: Elaborado pelo autor.

3.3 Implementação do modelo de inteligência artificial

Os dados utilizados para o treinamento foram obtidos do dataset CIC-IDS2017. O dataset foi obtido a partir do trabalho de Sharafaldin, Lashkari e Ghorbani (2018) e foi disponibilizado publicamente pelo *Canadian Institute for*

Cybersecurity (CIC) da universidade canadense *University of New Brunswick* (UNB). (CIC, 2025)

Para a implementação do modelo, o dataset foi dividido em três conjuntos, de treino, teste e validação, que correspondem a 70%, 15% e 15% da base de dados. Por conta do número desbalanceado de entradas para cada classe foi necessário fazer o uso do *RandomUnderSampler*, para remover o excesso da classe majoritária e o *SMOTE* para aumentar o número das classes minoritárias no conjunto de treinamento.

Levando em consideração que os dados para treinamento são rotulados, o método de treinamento mais apropriado é o supervisionado. com isso em mente, o modelo implementado foi um classificador baseado no framework open source PyTorch (2025).

O modelo foi criado em um container sequencial utilizando a função nn.Sequential e dentro do container foram usadas as seguintes instruções:

a) nn.Linear(inputDim, 256) em que inputDim é o número de features inicial da base de dados, consistindo de 78 features;
b) nn.ReLU() camada de inicialização que introduz não linearidade;
c) nn.BatchNorm1d(256) para realizar a normalização;
d) nn.Dropout(0.3) para zerar aleatóriamente 30% das unidades durante o treino para evitar *overfitting*;
e) nn.Linear(256, 128) outra camada de transformação linear que reduz a dimensionalidade de 256 para 128;
f) nn.ReLU() outra função ReLu para introduzir não linearidade;
g) nn.BatchNorm1d(128) outra função para normalização, mas agora para 128 features;
h) nn.Dropout(0.3) novamente zera aleatoriamente 30% das unidades durante o treinamento para evitar *overfitting*;
i) nn.Linear(128, numClasses) camada final que produz numClasses saídas, no caso, numClasses corresponde ao número de ataques presentes na base de dados, totalizando 15 classes.

Como o objetivo do classificador é diferenciar entre quinze classes, o algoritmo de cálculo para o erro foi o *CrossEntropyLoss*, que é voltado para problemas de classificação com n classes, além, da possibilidade de atribuir peso ao

erro de cada classe, o que foi necessário para diminuir o número de falsos positivos, característica particular da detecção baseada em anomalia. Para diminuir o número de falsos positivos foi definido o peso 3 para o erro na classe benigna, também foi realizada a normalização da escala de *loss* para que a média de aproximasse de 1.

O modelo foi otimizado com Adam com uma *learning rate* de 0,001 e um *weight decay* de 0,0001, para penalizar erros muito grandes e reduzir a possibilidade de *overfitting*. O treino foi concluído com 20 épocas utilizando um *batch size* de 64, vale ressaltar que foi feita uma função de treinamento do modelo para que o treino parasse antes de causar um *overfitting*. Durante o treino apenas o *state_dict* da iteração do modelo com o menor valor de *loss* foi salvo, com um limite de 20 iterações subsequentes (sem a queda do valor de l*oss*) para o fim do treino, no caso, a versão final do modelo apresentou o menor valor de l*oss* na época 20.

Para evitar que outliers na base de dados influenciassem o treinamento, foi necessário o uso de um escalonador. Foi selecionado o *RobustScaler,* que faz parte da biblioteca scikit-learn (2025), trabalhando o escalonamento das diferentes características dos dados de maneira independente de acordo com a área entre quartis.

3.4 Sistema

O sistema consiste em um com software de análise de rede que tem a capacidade de capturar os pacotes da rede e interpretá-los, ou seja, seja capaz de capturar o tráfego de rede e extrair informações dos pacotes capturados. Para diminuir o escopo do software, foi decidido trabalhar apenas com pacotes IPv4.

O software inicia o ciclo de captura até o limite de 500 pacotes capturados. Após o término da captura, os pacotes são interpretados e é gerado um fluxo de dados que indica conexões entre os ips da rede e as suas respectivas características. Vale ressaltar que o software foi desenvolvido com a plataforma windows em mente, com isso, não foi possível fazer o uso de bibliotecas para o cálculo do fluxo de dados, já que o software *CICFlowMeter* está disponível apenas para linux, para que os dados resultantes do processamento dos pacotes capturados fossem compatíveis com o dataset utilizado para o treino, as respectivas funções de cálculo dos parâmetros foram implementadas do zero.

Tendo extraído o fluxo de dados, o software pode então iniciar o modelo de inteligência artificial e realizar a predição. Ao término da operação o usuário é levado a página de resultados, onde são mostrados todos os fluxos identificados como maliciosos, assim como o log com todos os fluxos identificados como maliciosos em capturas anteriores.

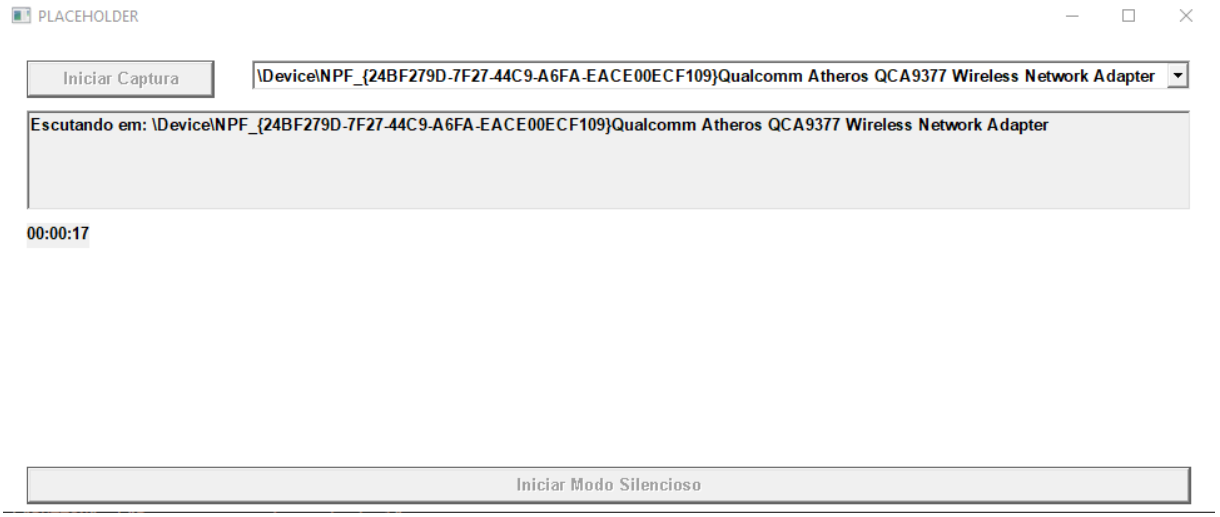
Dada a implementação do sistema, o modelo de inteligência artificial pode ser trocado sem que essa mudança crie a necessidade de fazer alterações no sistema principal, assim apenas a thread de processamento de pacotes deve ser alterada para que fique compatível com o modelo.

3.5 Telas do sistema

Com o objetivo de simplificar o uso do software, o sistema foi elaborado para que o usuário tenha o que realizar o menor número de ações para iniciar o funcionamento do software. A página inicial do software possui uma caixa de seleção para que o usuário possa identificar qual interface de rede o software deve utilizar para capturar os pacotes.

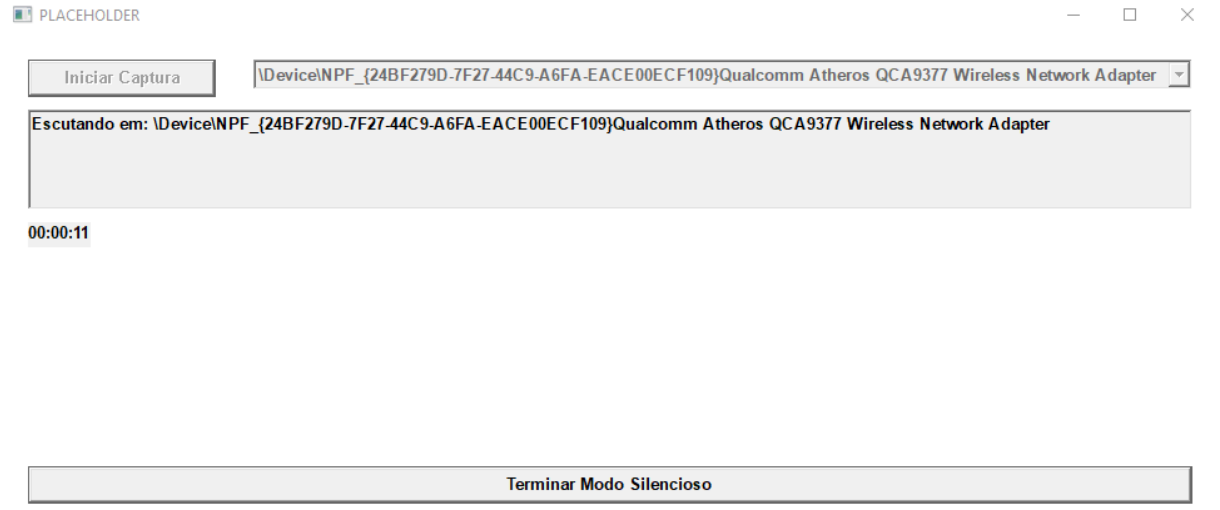
Com a interface selecionada o usuário tem duas opções, o botão “Iniciar Captura” que iniciará o funcionamento padrão do software (Figura 3). Outra opção possível é o botão “Modo Silencioso” que inicia o software de modo que ele não avise o usuário que terminou o ciclo de funcionamento e apenas reinicie a captura, de modo que o software continue realizando ciclos de captura e predição até que o usuário selecione o botão “Terminar Modo Silencioso” (Figura 4).

Figura 3 - Tela principal do sistema



Fonte: Elaborado pelo autor.

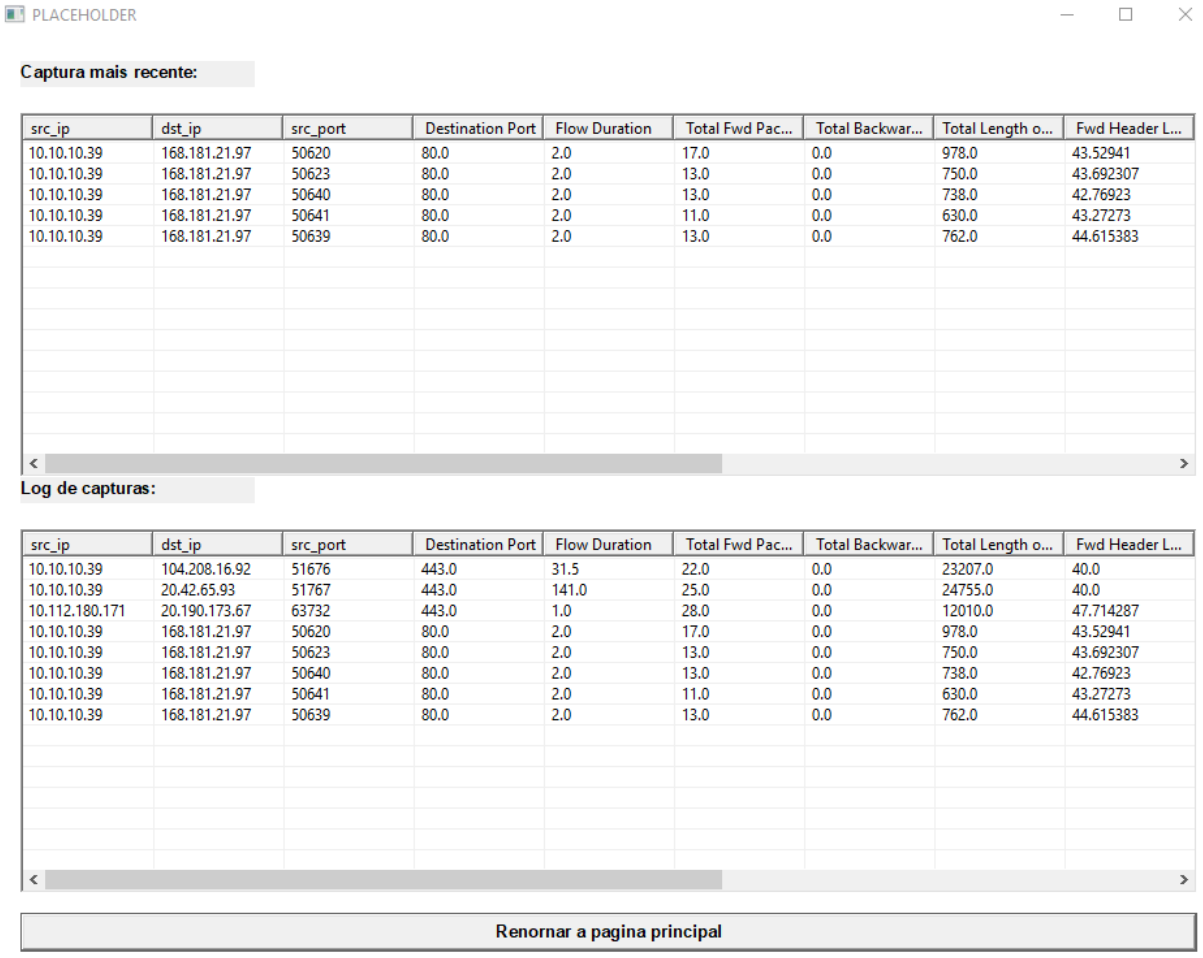
Figura 4 - Terminar o Modo silencioso



Fonte: Elaborado pelo autor.

Independente do modo de funcionamento selecionando,ao final da operação o software leva o usuário a página de resultados que contém os fluxos de dados identificados na última captura como maliciosos, além de um log contendo todos os fluxos identificados como maliciosos em capturas anteriores.(Figura 5)

Figura 5 - Tela de resultados



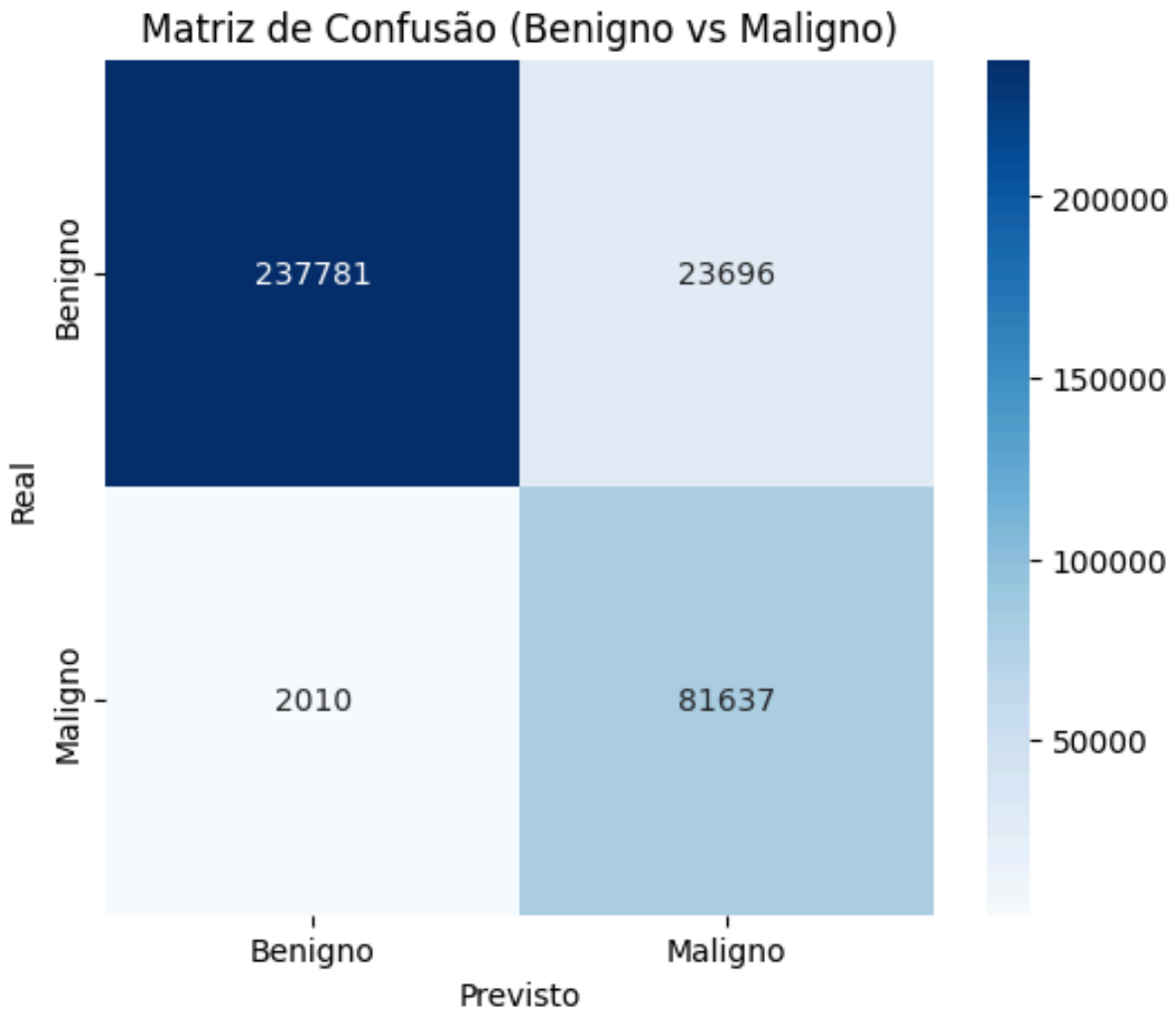
Fonte: Elaborado pelo autor.

4 PROCESSO DE VALIDAÇÃO

O processo de validação foi realizado utilizando o conjunto de dados de validação, que foi separado antes do treinamento do modelo. Como o objetivo do software não é classificar os tipos de ataque, mas sim, encontrar fluxos de dados maliciosos, as métricas de precisão foram calculadas de acordo com essa decisão, além de permitir melhor compreensão do desempenho do software em diferenciar fluxos malignos de benignos.

A imagem a seguir (Figura 6), mostra a matriz de confusão que indica a relação entre os rótulos reais e os rótulos preditos pelo modelo. Ao realizar cálculos com os valores apresentados podemos obter as métricas de precisão do modelo.

Figura 6 - Matriz de confusão



Fonte: Elaborado pelo autor.

O modelo apresenta um *recall* de 97,6 %, uma precisão para benignos de 77,5%, uma acurácia de 90,8%, uma taxa de falsos positivos de 9,06%, uma taxa de falsos negativos de 2,4% e um F1-score de 0,864. Isso significa que o modelo mantém uma boa proporção entre precisão e recall, dando prioridade ao recall, já que falsos negativos são muito mais perigosos, porém, isso causa um aumento na incidência de falsos positivos. Mesmo com o número de falsos positivos, o modelo permanece com a acurácia acima de 90%, o que significa que a chance de uma predição estar correta é de 90%. A tabela a seguir (Tabela 1) mostra as métricas de precisão do modelo para melhor visualização.

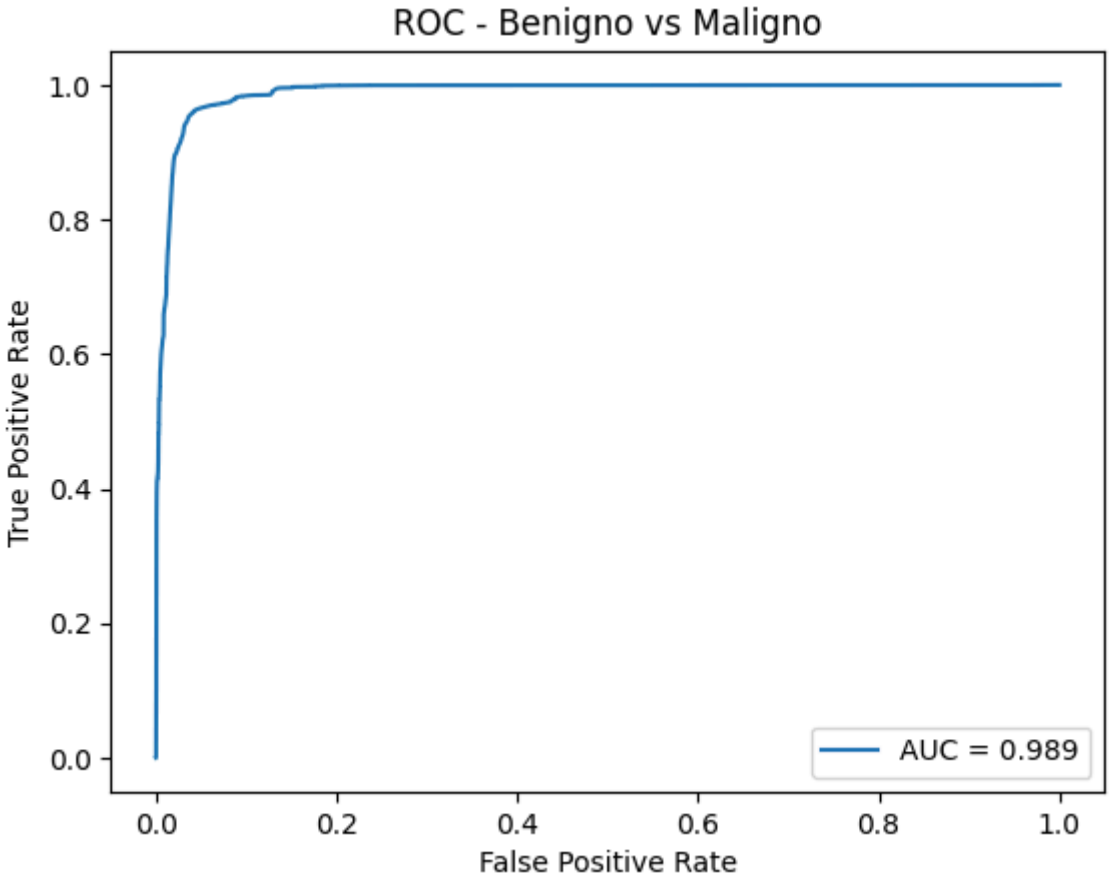
Tabela 1 - Tabela das métricas de precisão do modelo

Métrica	**Valor**	**Porcentagem (%)**
F1-score	0,864	-
Precisão (benignos)	0,775	77,5%
Recall	0,976	97,6%
Falsos positivos	0,0906	9,06%
Falsos Negativos	0,024	2,4%
Acurácia	0,908	90,8%
AUC	0.989	98,9%

Fonte: Elaborado pelo autor.

A imagem em sequência (Figura 7) mostra a curva ROC e o AUC, que mede a área sob a curva. Esse gráfico mostra a relação entre a sensibilidade e a taxa de falsos positivos em diferentes thresholds (limiares de decisão) do modelo, o formato da curva indica que o modelo tem uma alta sensibilidade e um baixo número de falsos positivos. A métrica AUC indica a chance do modelo atribuir um valor mais alto a uma amostra maligna do que uma amostra benigna, ambas escolhidas aleatóriamente, numericamente falando, a chance é de 98.9%.

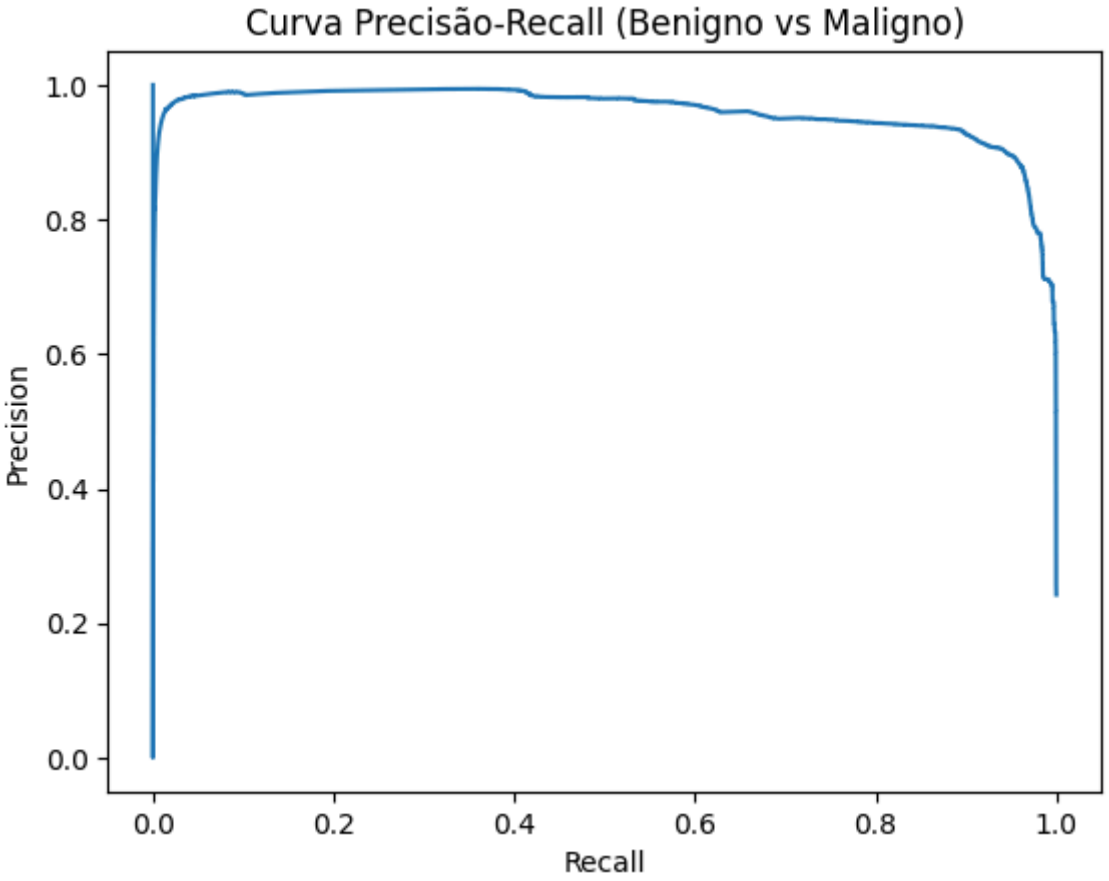
Figura 7 - Curva ROC AUC



Fonte: Elaborado pelo autor.

A imagem seguinte (Figura 8) mostra a curva de precisão-*recall*. Ela oferece um olhar mais sensível ao desbalanceamento de classes, o que não é possível com a ROC AUC, sendo é muito importante para evitar erros por conta da grande diferença numérica entre as classes. A curva de precisão-recall mostra a relação entre precisão e recall a medida que o limiar de decisão do modelo varia, na imagem podemos perceber que a precisão se mantém acima de 0.9 na maior parte da curva, e o recall também se aproxima de 1, o que significa que o modelo mantém uma alta precisão e um alto recall simultaneamente, sendo ideal para a o objetivo do software.

Figura 8 - Curva Precisão-*Recall*



Fonte: Elaborado pelo autor.

Com a análise do processo de validação, podemos concluir que o modelo oferece bom desempenho para o objetivo proposto para este trabalho. Em conjunto esses gráficos mostram que o modelo oferece ótima detecção de ataques com poucos falsos positivos, o que é necessário para a detecção de tráfego de rede anómalo.

5 CONCLUSÃO

O projeto iniciou-se com a intenção de criar uma ferramenta adicional que possa ajudar profissionais da área de segurança a identificar possíveis intrusões na rede. Com isso em mente é possível dizer que o software atingiu seus objetivos iniciais.

O software irá fornecer IPs na rede em que possivelmente estão transmitindo fluxos malignos de dados. Com isso, profissionais da segurança de informação podem fazer o uso de ferramentas mais complexas com o intuito de eliminar a ameaça da rede.

O fato do software ser desenvolvido com a devida separação de threads em mente facilita a substituição do modelo de predição em futuras melhorias, já que a substituição do modelo de predição impacta diretamente na efetividade de detecção.

Um dos principais desafios encontrados durante a elaboração do software foi o treinamento do classificador. Uma das principais razões sendo a dificuldade de balancear a porcentagem de falsos positivos.

5.1 Trabalhos futuros

Trabalhos futuros podem envolver a execução do software em um ambiente controlado para que seja possível verificar qual o seu desempenho em uma rede comprometida, a fim de confirmar sua eficácia em detectar dispositivos que estão comprometidos.

A nível de software também é possível aumentar a complexidade de funções disponíveis para que profissionais da área de segurança tenham uma maior flexibilidade na utilização do programa.

REFERÊNCIAS

TEAM KASPERSKY. **Kaspersky: mais de 400 mil malware foram descobertos por dia em 2023**. 2024. Disponível em: https://www.kaspersky.com.br/about/press-releases/2024_kaspersky-mais-de-400-mil-malware-foram-descobertos-por-dia-em-2023. Acesso em: 21 mar. 2024.

KASPERSKY. **Relatório da Kaspersky mostra principais ameaças com IoT**. 2023. Disponível em: https://www.kaspersky.com.br/about/press-releases/relatorio-da-kaspersky-mostra-principais-ameacas-com-iot. Acesso em: 27 out. 2025.

HAO, Guodong; XU, Ke; XU, Lei; WU, Bo. Detecting APT Malware Infections Based on Malicious DNS and Traffic Analysis. **Ieee Access: The journal for rapid open access publishing.** Beijing, p. 1132-1142. jun. 2015. Disponível em: https://ieeexplore.ieee.org/abstract/document/7163279. Acesso em: 27 mar. 2024.

TEAM KASPERSKY. **Cibercriminosos criam 400 mil novos malware por dia**. 2021. Disponível em: https://www.kaspersky.com.br/blog/cibercriminosos-criam-400-mil-novos-malware-por-dia/20423/. Acesso em: 04 abr. 2024.

KASPERSKY. **Cibersegurança para residências e empresas | Kaspersky Brasil.** Disponível em: https://www.kaspersky.com.br/ Acesso em: 30 jun. 2024.

NORTON. **Site oficial | Norton™ — Software antivírus e antimalware.** Disponível em: https://br.norton.com/# Acesso em: 30 jun. 2024.

BITDEFENDER. **Bitdefender - Líder global em software de segurança cibernética**. Disponível em: https://www.bitdefender.com.br/ Acesso em: 30 jun. 2024.

SNORT. **Snort - Network Intrusion Detection & Prevention System**. Disponível em: https://www.snort.org/ Acesso em: 30 jun. 2024.

WIRESHARK. **Wireshark • Go Deep**. Disponível em: https://www.wireshark.org Acesso em: 30 jun. 2024.

LABORATORY, Lincoln. **1999 DARPA INTRUSION DETECTION EVALUATION DATASET**. 1999. Disponível em: https://www.ll.mit.edu/r-d/datasets/1999-darpa-intrusion-detection-evaluation-dataset. Acesso em: 04 abr. 2024.

STALLINGS, William. **Criptografia e segurança de redes: princípios e práticas.** 4. ed.São Paulo: Pearson Education do Brasil, 2008.

HINTZBERGEN, Jule et al. **Fundamentos de Segurança da Informação: com base na ISO 27001 e na ISO 27002.** Rio de Janeiro: Editora Brasport, 2018. 201 p.

NPCAP. **Npcap: Windows Packet Capture Library & Driver**. Disponível em: <https://npcap.com/> Acesso em: 30 jun. 2024.

MICROSOFT. **Índice de API do Windows - Win32 apps | Microsoft Learn**.Disponível em: https://learn.microsoft.com/pt-br/windows/win32/apiindex/windows-api-list Acesso em 04 jun. 2025.

CIC. **IDS 2017 | Datasets | Research | Canadian Institute for Cybersecurity | UNB**. Disponível em: https://www.unb.ca/cic/datasets/ids-2017.html Acesso em 05 dez. 2025.

SHARAFALDIN, Iman; LASHKARI, Arash Habibi; GHORBANI, Ali A.. Toward Generating a New Intrusion Detection Dataset and Intrusion Traffic Characterization. **Proceedings Of The 4Th International Conference On Information Systems Security And Privacy**, Funchal, v. 1, n. 10, p. 108-116, jan. 2018. Disponível em: https://www.scitepress.org/PublicationsDetail.aspx?ID=K3WXGO8/3O4=&t=1. Acesso em: 05 dez. 2025.

PYTORCH. **torch — PyTorch 2.9 documentation**. Disponível em: https://docs.pytorch.org/docs/stable/torch.html Acesso em 08 de jun 2025

SCIKIT-LEARN. **scikit-learn: machine learning in Python — scikit-learn 1.7.2 documentation**.Disponível em: https://scikit-learn.org/stable/api/sklearn.preprocessing.html Acesso em 08 jun 2025