UNIVERSIDADE FEDERAL DE PERNAMBUCO

Universidade Federal de Pernambuco
Centro de Tecnologia e Geociências
Departamento de Eletrônica e Sistemas

Graduação em Engenharia Eletrônica

Matheus de Oliveira Roma

APLICAÇÃO DE MÁQUINAS MORFOLÓGICAS DE APRENDIZADO EXTREMO À DETECÇÃO DE MALWARES DO TIPO MEDIYES

Recife
2025

Matheus de Oliveira Roma

APLICAÇÃO DE MÁQUINAS MORFOLÓGICAS DE APRENDIZADO EXTREMO À DETECÇÃO DE MALWARES DO TIPO MEDIYES

Trabalho de Conclusão apresentado ao Curso de Graduação em Engenharia Eletrônica, do Departamento de Eletrônica e Sistemas, da Universidade Federal de Pernambuco, como requisito parcial para obtenção do grau de Bacharel em Engenharia Eletrônica.

Orientador(a): Prof. Sidney Marlon Lopes de Lima, D.Sc.

Recife
2025

Ficha de identificação da obra elaborada pelo autor,
através do programa de geração automática do SIB/UFPE

Roma, Matheus de Oliveira.
Aplicação de máquinas morfológicas de aprendizado extremo à detecção de malwares do tipo mediyes / Matheus de Oliveira Roma. - Recife, 2025.
60 p. : il., tab.

Orientador(a): Sidney Marlon Lopes de Lima
Trabalho de Conclusão de Curso (Graduação) - Universidade Federal de Pernambuco, Centro de Tecnologia e Geociências, Engenharia Eletrônica - Bacharelado, 2025.
Inclui referências.

1. Malwares. 2. Mediyes. 3. Máquinas Morfológicas de Aprendizado Extremo. 4. Segurança digital. I. Lima, Sidney Marlon Lopes de. (Orientação). II. Título.

000 CDD (22.ed.)

Matheus de Oliveira Roma

APLICAÇÃO DE MÁQUINAS MORFOLÓGICAS DE APRENDIZADO EXTREMO À DETECÇÃO DE MALWARES DO TIPO MEDIYES

Trabalho de Conclusão apresentado ao Curso de Graduação em Engenharia Eletrônica, do Departamento de Eletrônica e Sistemas, da Universidade Federal de Pernambuco, como requisito parcial para obtenção do grau de Bacharel em Engenharia Eletrônica.

Aprovado em: 20/03/2025

Banca Examinadora

Prof. Sidney Marlon Lopes de Lima, D.Sc.
Universidade Federal de Pernambuco

Prof. João Marcelo Xavier Natário Teixeira, D.Sc.
Universidade Federal de Pernambuco

Dedico este trabalho à minha namorada Karolyne e aos meus amigos Vitor e Ricardo, que me apoiaram e motivaram durante todo o processo

Agradecimentos

Gostaria de expressar minha sincera gratidão a todas as pessoas que, de alguma forma, contribuíram para a realização deste trabalho.

Em primeiro lugar, agradeço ao meu orientador, Prof. Sidney Lima, pela paciência, dedicação e orientação ao longo de todo o processo. Suas críticas e sugestões foram fundamentais para o desenvolvimento e aprimoramento deste trabalho.

Aos professores do curso, que compartilharam conhecimentos e experiências que enriqueceram minha formação acadêmica e profissional.

Aos meus amigos Vitor e Ricardo, pelo apoio técnico na revisão do trabalho e, principalmente, pela motivação e cobrança nos momentos em que mais precisei. Vocês foram peças-chave para que eu conseguisse finalizar este projeto dentro do prazo.

Aos colegas de turma, com quem compartilhei desafios, aprendizados e momentos de descontração ao longo dessa jornada.

Este trabalho é o resultado não apenas do meu esforço individual, mas também do apoio e contribuição de cada uma dessas pessoas. A todos, o meu muito obrigado.

Que este trabalho seja um reflexo não apenas do meu esforço, mas também da rede de apoio que me cercou.

Você deve aproveitar os pequenos desvios ao máximo. Porque neles você encontrará coisas mais importantes do que aquilo que você deseja.

Yoshihiro Togashi

Resumo do Trabalho de Conclusão de Curso apresentado ao Departamento de Eletrônica e Sistemas, como parte dos requisitos necessários para a obtenção do grau de Bacharel em Engenharia Eletrônica(Eng.)

APLICAÇÃO DE MÁQUINAS MORFOLÓGICAS DE APRENDIZADO EXTREMO À DETECÇÃO DE MALWARES DO TIPO MEDIYES

Matheus de Oliveira Roma

Este trabalho propõe a aplicação de Máquinas Morfológicas de Aprendizado Extremo na detecção de *malwares*, com foco no *Mediyes*. *Malwares* são programas projetados para acessar dispositivos sem autorização, representando uma ameaça à segurança digital. O *Mediyes* se destaca por empregar estratégias avançadas, como o uso de certificados digitais válidos e a capacidade de se infiltrar em sistemas comprometidos, interceptando buscas em motores populares e redirecionando-as para servidores maliciosos, gerando cliques fraudulentos e ganhos ilícitos. A contribuição deste trabalho é um antivírus autoral, desenvolvido com uma máquina de aprendizagem extrema e dotada de *kernels* morfológicos, oferecendo uma solução eficaz para o controle preventivo do *Mediyes*. Os resultados indicam um desempenho médio de 99,25% na distinção entre aplicativos benignos e o malware, com tempo médio de treinamento de 0,5 segundos, demonstrando eficiência e agilidade. A proposta contribui para a segurança digital, mitigando os riscos associados ao Mediyes.

Palavras-chave: *Malwares*, *Mediyes*, Máquinas Morfológicas de Aprendizado Extremo, Segurança Digital.

Abstract of Course Conclusion Work, presented to Departament of Eletronic and Systems, as a partial fulfillment of the requirements for the degree of Bachelor of Electronic Engineering(Eng.)

APPLICATION OF MORPHOLOGICAL EXTREME LEARNING MACHINES FOR DETECTION OF MEDIYES TYPE MALWARE

Matheus de Oliveira Roma

This work presents an antivirus that uses extreme learning machines. And special kernels to detect malware, with a focus on Mediyes. Malware, such as Mediyes, invades devices without authorization. Mediyes is particularly harmful. It uses valid digital certificates and targets compromised systems. It redirects search engine queries to malicious sites. This tactic generates fraudulent clicks and profits for the attackers. Our main contribution is an antivirus designed to combat Mediyes. It uses advanced technology to prevent the harmful effects of Mediyes. We have achieved an average detection rate of 99.25% for our antivirus. It distinguishes between safe applications and Mediyes. What's more, it trains in 0.5 seconds, demonstrating its efficiency. Our work leads to a great increase in digital security. It offers a promising solution against Mediyes malware threats.

Keywords: Malwares, Mediyes, Pseudo-Morphological Extreme Learning Machines, Digital Security.

Lista de Ilustrações

Lista de Tabelas

Lista de Símbolos

$\vee$	Operação lógica de "OU"(disjunção)
$\wedge$	Operação lógica de "E"(conjunção)
$\in$	. Simbolo de "pertence a"Algébrico
$\mathbb{N}$	.. Conjunto dos números naturais
$\prod$	..... Produtório Matemático
$\bigcap$	.. Interseção entre dois conjuntos
$\bigcup$	.... União entre dois conjuntos
$\mathbb{R}$	.... Conjunto dos números reais
$\dagger$	..... Adjunto de Hermitiano

Lista de Abreviações

ELM	Extreme Learning Machine
MLP	Multilayer Perceptron
mELM	Morphological ELM
UID	Unique Identifier
UC	Unidade de Controle
IoT	Internet of Things
DL	Deep Learning
GPU	Graphics Processing Unit
CUDA	Compute Unified Device Architecture
RAM	Random access memory
TLS	Transport Layer Security
API	Application Programming Interface
GUI	Graphical User Interface
DNS	Domain Name System
FTP	File Transfer Protocol
HTTP	Hypertext Transfer Protocol

Sumário

Capítulo 1

Introdução

O avanço acelerado da tecnologia tem transformado a dinâmica do acesso a informações na sociedade contemporânea. Essa evolução singular proporcionou inúmeros benefícios, agilizando e simplificando a oferta de serviços. Porém, esse progresso também abriu portas para indivíduos mal-intencionados explorarem, por meio da tecnologia, dados alheios, incluindo informações sigilosas de grande relevância para os usuários.

Nesse cenário, os *malwares* (corruptela entre *malicious + software*) emergem como uma das principais ameaças à segurança digital. Esses programas maliciosos são projetados para infiltrar, danificar ou roubar dados de sistemas computacionais. Os *malwares* podem se manifestar de diversas formas como vírus, cavalos de troia, *ransomwares*, *spywares*, entre outros. Vírus, por exemplo, são códigos maliciosos que se replicam e infectam outros arquivos, enquanto cavalos de troia disfarçam-se como software legítimo para enganar usuários e abrir portas para ataques. Já os *ransomwares* criptografam dados e exigem pagamento para liberá-los, causando prejuízos financeiros e operacionais significativos.

Diante dessa variedade de ameaças, os mecanismos de defesa digital evoluíram para combater não apenas vírus, mas todo tipo de *malware*. Embora o termo "anti-vírus" tenha se massificado e seja amplamente utilizado para descrever ferramentas de proteção, a nomenclatura mais precisa e abrangente seria "anti-malware", já que essas soluções visam detectar e neutralizar uma gama mais ampla de programas

maliciosos. No entanto, por razões históricas e de familiaridade do público, o termo "antivírus" permanece em uso comum.

Nesse contexto, destaca-se a ameaça representada pelo *Mediyes*, um *malware* que, ao obter certificados digitais válidos, infiltra-se em sistemas comprometidos, interceptando solicitações de pesquisa e redirecionando-as para servidores maliciosos. Esse comportamento fraudulento gera cliques não autorizados, resultando em ganhos ilícitos para os perpetradores.

Diante desse cenário, é fundamental explorar abordagens inovadoras e eficazes para conter o *Mediyes* e mitigar os riscos associados a esse tipo de ameaça digital. Este trabalho se propõe a investigar o uso de Máquinas Morfológicas de Aprendizado Extremo no desenvolvimento de um anti-malware, apresentando uma alternativa criativa e eficaz para enfrentar os desafios impostos por esse *malware* sofisticado.

Essa pesquisa visa contribuir significativamente para a segurança digital, oferecendo uma solução inteligente e eficaz para proteger os usuários contra os perigos do *Mediyes* e suas atividades prejudiciais. Para isso, são empregadas técnicas avançadas de ciência de dados, inteligência artificial e detecção de ameaças. A intenção é prover uma abordagem proativa na identificação e mitigação de riscos. Espera-se que os resultados desta pesquisa possam ser aplicados em diversas plataformas digitais, contribuindo para um ambiente virtual mais seguro e resiliente.

Após identificar a necessidade de um sistema de proteção mais eficiente, torna-se essencial realizar uma análise crítica do desempenho dos antivírus comerciais disponíveis no mercado, os quais apresentam significativas limitações na detecção de certos tipos de *malware*. A principal limitação dos antivírus atuais reside em seu método de reconhecimento, que se baseia em *blocklists* [1]. Nessa abordagem, o sistema mantém uma lista de vírus em sua base de dados e compara os arquivos suspeitos com essa lista. Vale ressaltar que cada antivírus utiliza um modelo próprio

[1] O termo "*blocklist*" é utilizado neste trabalho em substituição ao termo "*blacklist*", que, embora historicamente comum na área de segurança da informação, tem sido evitado devido às suas conotações pejorativas e à associação indireta com estereótipos raciais. A adoção de "*blocklist*" reflete uma tendência atual na área de tecnologia de promover uma linguagem mais inclusiva e neutra, alinhada com práticas de diversidade e equidade.

para armazenar essas listas, como mostrará a Tabela 1.1. Isso ocorre porque, devido a disputas comerciais, essas listas não são compartilhadas entre diferentes softwares, resultando na ausência de um padrão comum na catalogação dos *malwares*.

Por isso, é essencial considerar outro método de defesa que, ao invés de buscar assinaturas em *blocklists*, deve procurar correspondências de padrões entre *malwares* usando técnicas de inteligência computacional, como as redes neurais artificiais. Tecnicamente, as redes neurais especializadas na detecção de *malwares* são capazes de identificar comportamentos previamente classificados como suspeitos com uma acurácia média superior a 98% (LIMA et al., 2021). Dessa forma, o *malware* pode ser identificado de forma preventiva, mesmo antes de ser executado (clicado) pelo usuário. Como efeito colateral, o tempo de resposta das redes neurais é geralmente elevado. Se o tempo de resposta do mecanismo de cibervigilância for alto, um software malicioso que explore novas vulnerabilidades pode causar danos irreversíveis e irrecuperáveis em toda a internet.

Redes neurais particularmente grandes, como o Perceptron Multicamadas (MLP), exigem o conhecimento dos parâmetros da rede para obter o máximo desempenho na resolução do problema (LIMA et al., 2021). Uma preocupação comum nesse tipo de rede é evitar a aderência a mínimos locais de análise, sendo necessário adicionar métodos de controle da rede para se desprender dessas regiões (LIMA et al., 2021). Outra característica comum desse tipo de rede é o elevado tempo de treinamento necessário para torná-la capaz de realizar classificações corretas.

A rede ELM (*Extreme Learning Machine* - Máquina de Aprendizado Extremo) tem como principal caraterística a velocidade de treinamento e previsão de dados quando comparada às redes neurais MLP. As ELMs são máquinas de aprendizagem poderosas e flexíveis baseadas em *kernel*, cujas principais características são o treinamento rápido e o desempenho robusto de classificação. A rede ELM é uma rede de camada oculta única, não recorrente, baseada num método analítico para estimar os pesos de saída da rede, em qualquer inicialização aleatória dos pesos referentes às ligações sinápticas entre os neurônios. As redes ELM têm sido amplamente apli-

cadas numa grande variedade de áreas, como a engenharia biomédica (LIMA et al., 2021) (LIMA et al., 2020) (LIMA et al., 2016) (PEREIRA, 2020) (AZEVEDO and *et al.*, 2015) (AZEVEDO and *et al.*, 2020). As redes ELM podem contribuir muito para a melhoria da segurança de diversos dispositivos.

O trabalho apresentado aplica ELMs na área de segurança da informação, em particular na identificação de padrões de *malware* do tipo *Mediyes*. As propriedades dos arquivos analisados servem como parâmetros de entrada para as redes neurais artificiais, para que estas sejam utilizadas como agentes de classificação. O intuito é classificar aplicações consideradas suspeitas em duas classes: benignas ou *malwares*. A aprendizagem ELM é fundamentada em *kernels*.

No lugar dos *kernels* convencionais, são usados os mELMs (*Morphological* ELMs), ELMs com *kernels* de camadas ocultas os quais são inspirados em dois operadores morfológicos do processamento de imagens: Erosão e Dilatação. No que diz respeito às experiências, os resultados são comparados com os antivírus mais avançados do mercado e avaliados através de métricas de classificação largamente utilizadas. Tais resultados são ainda comparados com os melhores cenários obtidos pelo estado-da-arte. O antivírus autoral atinge um desempenho médio de 99,25% na diferenciação entre aplicações benignas e *Malware (Mediyes)*, com um tempo de treino de 0,55 segundos.

1.1 Limitações dos Antivírus Comerciais

A metodologia de defesa e resposta dos antivírus comerciais tem por base o fato de que os arquivos considerados suspeitos são ou não referenciados em registros chamados *blocklists*. Logo é possível que o *hash* do arquivo em análise não esteja na *blocklist* do antivírus para evitar a detecção do *malware*. Este código *hash* funciona como um UID (*unique identifier*) para um arquivo malicioso específico.

Em face das limitações dos antivírus comerciais, o desenvolvimento e a distribuição de variantes de aplicações maliciosas não é assim tão complicado. É relativamente simples, basta fazer ligeiras alterações no *malware* original fazendo uso de

subrotinas menos usuais, tais como laços ou condicionais sem escopo. Entretanto, devido a essas ligeiras alterações no *malware*, o *hash* modificado deste é diferente do *hash* do *malware* original. Consequentemente, o *malware* modificado acaba não sendo detectado pelo antivírus que classificou o arquivo nocivo original.

Vale a pena ainda a menção da existência de *exploits*, que são responsáveis pela criação e disseminação de variantes do *malware* original automaticamente. Portanto os antivírus baseados em *blocklists* acabam se mostrando ineficazes diante de qualquer variante de *malware* conhecido (SANS, 2017) (LIMA, 2020).

Visando fazer uma análise ainda mais precisa, o presente trabalho analisa a performance de 80 antivírus comerciais quanto à detecção de Mediyes. Utilizou-se, para isso, de dois processos distintos da plataforma VirusTotal: um deles sendo responsável pelo encaminhamento dos arquivos à análise nos servidores do VirusTotal, enquanto que o outro sendo responsável pela geração do diagnóstico dos antivírus analisados.

Foram analisadas 1.865 amostras *Mediyes* neste teste de detecção. Os resultados de performance são exibidos na Tabela 1.2, e seu diagnóstico pode ser categorizado de três formas distintas. A plataforma fornece um diagnóstico de *"malware"* sempre que o *software* detecta a presença de atividades maliciosas no arquivo suspeito, "benigno" nos casos em que o antivírus falha em identificar o *"malware"* e "omissão" nos casos em que o antivírus não diagnostica o arquivo malicioso.

A margem de detecção de *malware* pelos antivírus oscilou entre 0% e 98,98%. Em geral, eles foram capazes de detectar com acurácia 70,42% e um desvio padrão de 28,23%. O desvio padrão elevado evidencia que alguns programas de antivírus comerciais obtiveram êxito na detecção de *malwares*, ao passo que outros foram completamente ineficazes. Ou seja, apenas alguns dos programas de antivírus comerciais dispõem de uma *blocklist* grande o suficiente para prover uma elevada taxa de detecção de arquivos maliciosos que tenham sido catalogados anteriormente. Portanto o nível de proteção proporcionado por um antivírus contra uma ciberinvasão se relaciona diretamente ao volume e ao escopo de sua *blocklist*.

Também é importante mencionar que, em média, 17,80% dos antivírus erraram o diagnóstico de arquivos maliciosos, considerando-os como benignos (falsos negativos) tendo um desvio padrão de 19,87%. Adicionalmente, uma média de 11,78% dos antivírus sequer forneceram algum diagnóstico acerca dos arquivos (omissão) com um desvio padrão de 15,82%. Portanto essa taxa considerável de falsos negativos e omissões leva a um alto risco de infeção que pode ocasionar danos aos dados e computadores dos usuários, governos ou empresas, em grade parte de forma irreversível. Tais resultados apontam para a ineficácia dos antivírus comerciais em proteger os sistemas contra programas maliciosos em tempo real.

Para além das limitações mencionadas, os antivírus comerciais não fornecem informações úteis aos usuários sobre os arquivos maliciosos e, com frequência, arquivos maliciosos diferentes são apresentados apenas com nomes genéricos. Portanto é razoável afirmar que as informações de diagnóstico fornecidas pelos antivírus não servem para que o usuário compreenda o grau de nocividade do *malware* ou quais são suas consequências nocivas para o sistema, por exemplo. Isso quer dizer que, quando um mesmo arquivo maliciosos é analisado por dois programas antivírus diferentes, provavelmente eles darão classes completamente diferentes para o referido arquivo. Para além disso, duas variantes de um mesmo arquivo infetado, com alterações significativas no código, podem ser classificadas de duas formas completamente diferentes pelo mesmo programa de antivírus. Esta ausência de padrões na nomeação de amostras dificulta a implementação de soluções de cibersegurança, já que cada tipo de *malware* tem de ser tratado de forma específica.

Em conclusão, a inconsistência nos diagnósticos e a atribuição arbitrária de nomes pelos antivírus dificultam a identificação de arquivos maliciosos. Como indicado na Tabela 1.1, essa falta de padronização compromete a confiança dos diagnósticos. Por isso, não se pode esperar que uma técnica de aprendizagem automática consiga generalizar a detecção de arquivos maliciosos apenas com base nas informações fornecidas por antivírus comerciais.

Tabela 1.1: Resultados da submissão de dois malwares para o VirusTotal.

Antivirus	$VirusShare_A$	$VirusShare_B$
Cyren	W32/Mediyes.B.gen!Eldorado	W32/Mediyes.B.gen!Eldorado
MAX	malware (ai score=100)	malware (ai score=100)
Ad-Aware	Gen:Variant.Graftor.20936	Gen:Variant.Graftor.20936
MicroWorld-eScan	Gen:Variant.Graftor.20936	Gen:Variant.Graftor.20936
BitDefender	Gen:Variant.Graftor.20936	Gen:Variant.Graftor.20936
Webroot	W32.Trojan.Gen	W32.Trojan.Gen
GData	Win32.Rootkit.Mediyes.B	
DrWeb	Trojan.Mediyes.1	Trojan.Mediyes.1
Emsisoft	Gen:Variant.Graftor.20936 (B)	
ESET-NOD32	a variant of Win32/Mediyes.E	a variant of Win32/Mediyes.E
TACHYON	False Negative	False Negative
Malwarebytes	False Negative	False Negative
Paloalto	False Negative	False Negative
Avast-Mobile	False Negative	False Negative
Zoner	False Negative	False Negative
Bkav	HW32.Packed.	HW32.Packed.
Lionic	Trojan.Win32.Generic.lpZj	Trojan.Win32.Generic.4!c
CMC	Rootkit.Win32.Mediyes!O	Rootkit.Win32.Mediyes!O
CAT-QuickHeal	Trojan.Mediyes.B	Trojan.Mediyes.B
VIPRE	Trojan.Win32.Generic!BT	Trojan.Win32.Generic!BT

Fonte: Repositório de análise de malwares (Mediyes, 2024)).

1.2 Justificativa

A criação de um anti-malware especializado na detecção do *malware Mediyes* é essencial devido à sua capacidade de se infiltrar em sistemas, interceptar solicitações de pesquisa e redirecioná-las para servidores maliciosos, gerando cliques fraudulentos e lucros ilícitos. O *Mediyes* utiliza certificados digitais válidos para se disfarçar como software legítimo, dificultando sua identificação por métodos tradicionais baseados em *blocklists*.

A implementação de ELMs oferece uma abordagem inovadora para detectar padrões de comportamento do *Mediyes* antes que ele cause danos ao sistema. Diferente dos antivírus convencionais, as ELMs analisam características morfológicas e comportamentais, permitindo uma detecção proativa, mesmo contra variantes desconhecidas do *malware*.

Além de proteger os usuários contra infecções que podem resultar em perda

Tabela 1.2: Resultados dos antivírus comerciais.

Antivirus	Detecção (%)	Falso Negativo (%)	Omissão (%)
Cyren	98,98%	0,91%	0.11%
MAX	98,28%	1,55%	0.16%
Ad-Aware	98,28%	1,72%	0%
MicroWorld-eScan	98,18%	1,72%	0,11%
BitDefender	98,18%	1,77%	0,05%
Webroot	98,02%	1,82%	0,16%
GData	97,86%	1,77%	0,38%
DrWeb	97,8%	1,98%	0,21%
Emsisoft	97,69%	1,93%	0,38%
ESET-NOD32	97,64%	2,36%	0%
TACHYON	14,8%	84,99%	0,21%
Malwarebytes	14,16%	85,31%	0,54%
Paloalto	10,03%	89,92%	0,05%
TheHacker	8,1%	3,22%	88,69%
Gridinsoft	1,39%	23,32%	75,28%
AVware	0,16%	0%	99,84%
Avast-Mobile	0%	72,82%	27,18%
Zoner	0%	99,79%	0,21%
Babable	0%	11,05%	88,95%
Trustlook	0%	11,15%	88,85%

Fonte: Repositório autoral de análise de malwares (Mediyes, 2024).

de dados e degradação do sistema, este trabalho contribui para o avanço teórico na aplicação de técnicas de inteligência computacional na segurança da informação. Ao explorar operadores morfológicos como erosão e dilatação, propõe-se uma solução robusta e adaptável às ameaças digitais atuais.

1.3 Objetivo Geral

Desenvolver um anti-malware especializado, utilizando ELM, capaz de identificar o *malware Mediyes* com base na análise de padrões morfológicos e comportamentais, sem depender de métodos tradicionais como *blocklists*, proporcionando uma detecção proativa e eficiente.

1.4 Objetivos específicos

- Analisar as características e comportamentos do *malware Mediyes*, com foco em seu método de infiltração e redirecionamento de solicitações;
- Identificar as limitações dos métodos tradicionais de detecção de *malware*, como o uso de *blocklists*, no combate ao *Mediyes*;
- Implementar um protótipo de anti-malware baseado em ELMs para a detecção proativa do *Mediyes*;
- Avaliar o desempenho do protótipo desenvolvido, comparando-o com soluções de antivírus comerciais e técnicas de detecção baseadas em inteligência artificial.

1.5 Organização do TCC

O teor da presente tese divide-se em seis capítulos e em um apêndice. É possível verificar as referências em suas últimas páginas. A seguir, encontra-se um resumo dos seguintes capítulos.

Capítulo 2. Fundamentação Teórica, capítulo que introduz as Redes ELM, redes neurais de treinamento rápido e não iterativo, e sua variante morfológica (mELM), que aplica operadores de Erosão e Dilatação para melhor classificação de padrões. Discute-se sua vantagem sobre métodos tradicionais, como MLP, e sua aplicação em detecção de *malware*, destacando eficiência, ausência de *overfitting* e adaptabilidade a fronteiras complexas.

Capítulo 3. Estado da Arte, capítulo que analisa a evolução da detecção de *malware*, contrastando antivírus tradicionais com abordagens baseadas em redes neurais. Apresenta estudos de caso com altas taxas de acurácia, mas discute desafios como tempo de treinamento e escalabilidade.

Capítulo 4. Metodologia, capítulo que elucida os procedimentos, métodos, equipamentos e processo de classificação de arquivos utilizados na elaboração e execução do projeto proposto.

Capítulo 5. Resultados, capítulo onde são apresentados os valores obtidos do processo de treinamento e teste do proposto software antivírus baseado em rede neural.

Capítulo 6. Conclusão, parte final onde é apresentado um resumo dos principais pontos discutidos ao longo do trabalho, destacando os resultados mais importantes.

Capítulo 2

Fundamentação Teórica

2.1 Redes Neurais ELM

Os modelos de inteligência computacional conhecidos como redes neurais são amplamente empregados na resolução de tarefas de classificação, destacando-se pela capacidade de generalizar padrões mesmo quando expostos a dados nunca antes vistos. Em muitas dessas arquiteturas, como o MLP (XIANG et al., 2005), é essencial compreender e ajustar os parâmetros da rede para alcançar a melhor performance na resolução do problema. Um desafio recorrente nesse contexto é evitar que o modelo fique preso em mínimos locais (HUANG, 2000), o que exige a implementação de técnicas específicas para que a rede consiga escapar dessas regiões. Além disso, um aspecto frequentemente observado é o tempo considerável demandado pelo processo de treinamento, necessário para que a rede atinja um nível satisfatório de precisão nas classificações.

A ELM (Extreme Learning Machine ou Máquina de Aprendizado Extremo) se destaca pela rapidez no treinamento e na predição de dados quando comparada a outros modelos de classificação (HUANG, 2012). Essa arquitetura consiste em uma rede neural com apenas uma camada oculta e opera de forma não iterativa. O mecanismo de aprendizado da ELM utiliza a inversa generalizada de Moore-Penrose (também conhecida como pseudo-inversa) para calcular os pesos que conectam a camada oculta à camada de saída (HUANG, 2012).

O processo de aprendizado da ELM ocorre em modo de lote, ou seja, todos os dados são fornecidos à rede antes que os pesos das conexões sinápticas entre os neurônios da camada oculta e da camada de saída sejam ajustados. Diferente de outras redes neurais, a ELM realiza o aprendizado em uma única iteração, o que elimina a necessidade de retropropagação. Como resultado, a ELM não enfrenta problemas comuns como o overfitting (ou superajuste), que ocorre quando o modelo se adapta excessivamente aos dados de treinamento.

A ausência de iterações múltiplas torna o treinamento da ELM significativamente mais rápido do que métodos tradicionais. Outra vantagem é que não há necessidade de definir um número máximo de iterações, já que o algoritmo não é iterativo. Além disso, por não depender do método de gradiente descendente, a ELM não está sujeita a problemas como a convergência para mínimos locais nem exige a configuração de uma taxa de aprendizado.

Do ponto de vista matemático, na ELM os atributos de entrada x_{ik} são representados pelo conjunto $\{x_{it} \in \mathbb{R}; i \in \mathbb{N}^*, i = 1, ..., n; t \in \mathbb{N}*, t = 1, ..., v\}$. Esse conjunto contém n características extraídas da aplicação em questão, além de v vetores de dados utilizados para o treinamento. Já a camada oculta, composta por m neurônios, pode ser descrita pelo conjunto $\{h_j \in \mathbb{R}; j \in \mathbb{N}^*, j = 1, ..., m\}$.

O treinamento da ELM é conhecido por sua eficiência, já que envolve um número reduzido de etapas. O processo começa com a definição aleatória dos pesos de entrada w_{ji} e bias b_{jt}. Utilizando uma função de ativação $f : \mathbb{R} \to \mathbb{R}$, o aprendizado é realizado em três etapas principais:

1. Atribuição aleatória dos pesos: Os pesos w_{ji}, que conectam a camada de entrada à camada oculta, e os bias b_{jk} são inicializados de forma aleatória.

2. Cálculo da matriz H: Essa matriz representa as saídas dos neurônios da camada oculta.

3. Cálculo dos pesos de saída $\beta = H^{\tau}Y$, onde H^{τ} é a matriz inversa generalizada de Moore-Penrose da matriz H, e Y corresponde à matriz das saídas desejadas s.

Assim como no caso da MLP, a matriz H, que representa as saídas da camada

oculta, é calculada com base na função de ativação, das entradas e dos pesos da camada oculta, conforme descrito na Equação 2.1.

$$H_{jt} = \begin{bmatrix} K(11) & ... & K(1N) \\ ... & ... & ... \\ K(V1) & ... & K(VN) \end{bmatrix} \tag{2.1}$$

Diferente das redes que utilizam retropropagação, a ELM não requer a definição de critérios de parada para o treinamento nem a implementação de mecanismos para preservar a capacidade de generalização. Isso ocorre porque a ELM opera em uma única iteração, eliminando a necessidade de dividir os dados em conjuntos de treinamento, validação e teste. Basta separar os dados em conjuntos de treinamento e teste, o que permite alocar um número maior de amostras para esses dois grupos em comparação com redes neurais que dependem de retropropagação.

Após o treinamento, os padrões de teste são fornecidos à rede junto com as saídas esperadas. Nessa fase, a rede não realiza mais ajustes, limitando-se a calcular os resultados para cada conjunto de teste. A precisão da ELM é então avaliada comparando-se os valores esperados com os valores obtidos.

2.2 ELMs Morfológicas

O antivírus desenvolvido neste trabalho incorpora redes neurais do tipo ELM com o objetivo de identificar padrões de *malwares*. Em vez de utilizar *kernels* convencionais, o estudo propõe a criação de *kernels* autorais específicos para as ELMs. Neste artigo, são utilizadas as mELMs (ELMs morfológicas), uma variação das ELMs cujos núcleos da camada oculta são inspirados em operadores morfológicos de processamento de imagem, como Erosão e Dilatação. A proposta sugere que esses *kernels* morfológicos possuem a capacidade de se adaptar a qualquer tipo de fronteira de decisão.

A Morfologia Matemática é uma teoria abrangente de processamento não linear, amplamente empregada no tratamento de imagens digitais. Diversas aplicações es-

pecíficas derivam dessa teoria, incluindo detecção e segmentação de objetos, extração de características, entre outras (HUANG, 2012). A morfologia se baseia em transformações que alteram formas enquanto preservam as relações de inclusão entre os objetos. As duas operações morfológicas fundamentais são a Erosão e a Dilatação (HUANG, 2012). Essa teoria é considerada construtiva, uma vez que todas as suas operações são derivadas a partir desses dois conceitos básicos. Matematicamente, a Erosão e a Dilatação são definidas conforme as Equações 2.2 e 2.3, respectivamente:

$$\varepsilon_g(f)(u) = \bigcap_{v \in s} f(v) \vee \bar{g}(u - v) \tag{2.2}$$

$$\delta_g(f)(u) = \bigcup_{v \in s} f(v) \wedge g(u - v) \tag{2.3}$$

Nas equações, $f : S \to [0, 1]$ e $g : S \to [0, 1]$ representam imagens normalizadas, expressas como matrizes de formato S, onde $S \in \mathbb{N}^2$. Cada pixel é definido por um par cartesiano $(u, f(u))$, em que u indica a posição e $f(u)$ o valor associado. A matriz v corresponde a $f(u)$, delimitada por g. Os operadores $\bigcup$ e $\vee$ estão relacionados à operação de máximo, e os elementos $\bigcap$ e $\wedge$ estão associados à operação de mínimo. O elemento estruturante g é utilizado tanto para a Erosão quanto para a Dilatação (SANTOS, 2011), enquanto $\bar{g}$ representa a negação de g.

Na Equação 2.2, o processo começa com a negação do elemento estruturante $\bar{g}$. Em seguida, realiza-se a operação de máximo, representada por $f(v) \vee \bar{g}(u - v)$, onde $f(v)$ refere-se à matriz da imagem original f abrangida por $\bar{g}$, e $f(v)$ é formalmente chamada de região ativa da imagem. Por fim, o valor $\varepsilon_g(f)(u)$, na posição u da imagem erodida, recebe o mínimo entre os máximos calculados, utilizando o operador $\bigcap$. $\varepsilon_g(f)(u)$ assume 0, que corresponde ao preto absoluto. A Erosão sobrepõe $\bar{g}$ à imagem original f, com o objetivo de expandir áreas semelhantes a g (SANTOS, 2011). Associando 1 ao branco absoluto e 0 ao preto absoluto, a Erosão amplia as regiões mais escuras e remove áreas de maior intensidade (SANTOS, 2011).

O trabalho proposto introduz as mELMs, que são aplicadas, como estudo de

caso, na distinção entre executáveis benignos e *malwares*. Essas redes são inspiradas na Morfologia Matemática, utilizando os operadores não lineares de Erosão e Dilatação como base. A partir da Equação 2.2, que define o operador de Erosão, o *kernel* ELM de Erosão pode ser expresso conforme a Equação 2.4, onde $\{i \in \mathbb{N}^*, i = 1, ..., n; j \in \mathbb{N}^*, j = 1, ..., m; t \in \mathbb{N}^*, t = 1, ..., v\}$. Nessa configuração, a rede possui n neurônios na camada de entrada (excluindo o bias), m neurônios na camada oculta e v vetores de dados de treinamento, estabelecendo uma estrutura que combina eficiência e precisão para a tarefa proposta.

$$K_{\varepsilon}(t, i) = \bigcap_{i=1}^{n} (x_{it} \vee \bar{w_{ji}}) + b_{jt} \tag{2.4}$$

A implementação da mELM proposta neste trabalho, assim como os *kernels* morfológicos adaptativos, está disponível publicamente[1], permitindo a replicação dos experimentos e o uso pela comunidade acadêmica.

[1] mELM: `https://github.com/DejavuForensics/mELM`.

Capítulo 3

Estado da Arte

Ainda que venha sendo questionado por mais de uma década, o modus operandi dos antivírus segue sendo embasado em assinaturas de arquivos suspeitos quando estes são buscados em bancos de dados denominados como *blocklists* (LIMA, 2020)(SANS, 2017). Ou seja, o arquivo investigado é comparado ao *malware* catalogado na *blocklist*. Se esta não estiver atualizada, o *malware* não será identificado, o que acarretará em uma infecção ao dispositivo.

3.1 Limitações dos Antivírus Tradicionais e a Ascensão das Redes Neurais

Os antivírus tradicionais, baseados em assinaturas e *blocklists*, têm sido amplamente utilizados, mas apresentam limitações significativas, especialmente em um cenário onde novas variantes de *malware* surgem constantemente. Com o avanço das técnicas de aprendizado de máquina, as redes neurais emergiram como uma solução promissora, oferecendo maior acurácia e capacidade de generalização.

LIMA, *et al.* (2021) desenvolveram um antivírus capaz de remover *malwares* (Windows) com acurácia média de 98,32%. Dentro do sistema aplicado, para que a intenção nociva de um dado arquivo executável possa ser determinada, esse arquivo deve passar por um processo de desmembramento. Na execução do antivírus LIMA

et al. (2021), 630 recursos diferentes são coletados de cada arquivo executável. Esses dados são então utilizados como os neurônios de entrada da rede neural artificial. Esta classificação do antivírus distribui os arquivos de 32 bits em duas categorias: software inofensivo e software malicioso. A rede neural não é profunda.

Por outro lado, antivírus que utilizam redes neurais profundas alcançaram também elevadas taxas de acurácia. SU, J. *et al.* (2018) alcançaram uma acurácia média de 94,00% na detecção de *malware* em dispositivos de *IoT* (Internet das Coisas) (SU and VASCONCELLOS, 2018). Essa rede neural profunda tem uma estrutura de 6 camadas, 3 de aprendizagem com os seguintes pesos: 2 para camadas de dobragem e 1 camada inteiramente conectada. Esta rede ainda é treinada com um total de 5.000 iterações, um tamanho de lote de treino de 32 e uma taxa de aprendizagem de 0,0001.

FARUKI, P. *et al.* (2019) obteve uma média de 98,65% para a detecção de *malwares Android* (FARUKI and BUDDHADEV, 2019). FARUKI, P. *et al.* (2019) usa uma rede de aprendizagem profunda ReLU (Rectified Linear Unit), dotada de tecnologia de abandono. As camadas ReLU realizam operações de limiar para cada elemento da entrada, atribuindo a zero quaisquer valores que sejam inferiores a zero.

O antivírus desenvolvido por HOU, S. *et al.* (2016) visa combater *malwares* para sistemas de dispositivos *Android* através do uso de redes neural profundas (HOU and SAAS, 2016). A construção completa de uma pilha RBM (*Restricted Boltzmann Machine*) é o que compõe a rede neural profunda. A arquitetura é formada por 3 camadas ocultas com 200 nós em cada. HOU, S. *et al.* (2016) obtém uma acurácia média de 96,66%.

Apesar disso tudo, vale apontar que as redes profundas têm uma grande desvantagem: o tempo de aprendizagem é longo, pois cada camada é executada em sequência e, por isso, o número de ligações em paralelo é menor. Por conseguinte, só é possível efetuar uma alteração após a conclusão da execução da camada de cima. Quando se transpõe isso a aplicações que necessitem de treino constante, tal como o antivírus, um empecilho é encontrado, ainda mais se considerarmos a métrica de

criação de 8 novos *malwares* a cada segundo (Intel, 2018). Em resumo, o período de aprendizagem do *software* de antivírus deve ser superior ao ritmo de geração de novos programas maliciosos.

Dados os ótimos resultados adquiridos por técnicas de aprendizagem profunda, formou-se um senso comum de que esse método é capaz de conferir a maior acurácia a qualquer tipo de aplicação, o que não é verdade. Essas redes de aprendizagem profunda são construídas a partir de filtros convolucionais lineares, especialmente as redes convolucionais. Tais filtros atuam fundamentalmente em aplicações computacionais, mas costumam se tornar limitados quando aplicados onde são encontradas representações reais distintas em dados com valores parecidos.

Na análise de imagens de aparelhos de mamografia, por exemplo, frequentemente as imagens deste tipo sofrem de bastante ruído, o que torna difícil a identificação de uma lesão mamária. Esse é o tipo de caso em que os filtros convolucionais se mostram fundamentais na remoção do ruído e, logo, na eliminação de irregularidades no diagnóstico que poderiam corresponder a uma possível doença cancerígena. Técnicas de redução de ruído como essas são essenciais, e um exemplo bastante conhecido é o de filtros Gaussianos.

A título de contraexemplo, consideremos apenas uma parte dos dados apresentados na Tabela 3.1. Os pontos estão dispostos completamente isolados entre si, embora estejam na mesma região. O acesso à galeria de fotos ou ao *browser* de uma vítima não se relaciona a uma aplicação com suspeita de *scannear* de dados *WiFi*. Assim, se fosse aplicado um filtro de convolução linear ao acesso ao *browser* que contém o valor 0, o mesmo seria considerado ruído, visto que seus dados adjacentes apresentam valores positivos. Resumidamente, esta aplicação suspeita seria acusada de dar acesso ao *browser*, o que é um erro. Apesar disso, as técnicas de convolução enfrentam uma desvantagem em comparação às demais técnicas quando aplicadas à detecção de padrões de *malware*.

A fim de comprovar a fundamentação teórica, propõe-se um antivírus com uma rede neural morfológica superficial ao invés de redes convolucionais profundas. Expe-

rimentalmente, o antivírus apresenta uma acurácia comparável à da próxima geração de redes neurais superficiais e profundas. O antivírus proposto visa a combinação de alta acurácia com menor tempo de aprendizagem. A fim de prevenir comparações desleais, a etapa de seleção de recursos é normalizada monitorando 6.824 indicadores de comportamento que o arquivo suspeito possa executar caso sejam propositalmente ativados.

3.2 Redes Neurais na Detecção de *Malware*: Vantagens e Desafios

As redes neurais podem ser classificadas em rasas e profundas, cada uma com características distintas. Enquanto as redes rasas são mais simples e rápidas para treinar, as redes profundas oferecem maior capacidade de generalização, mas exigem mais dados e tempo de processamento. No entanto, o uso de *deep learning* na cibersegurança enfrenta desafios significativos, como a necessidade de grandes volumes de dados rotulados e a dificuldade de paralelização das redes profundas. Além disso, técnicas como a convolução, embora eficazes em aplicações como visão computacional, apresentam limitações quando aplicadas à detecção de *malware*.

Sem dúvida, os modelos de DL (*Deep Learning*) exibem capacidades impressionantes de generalização. Devido aos resultados excepcionais alcançados pelas técnicas de *deep learning*, surgiu uma noção predominante de que o aprendizado profundo pode oferecer acurácia superior em diversos tipos de aplicações. Em comparação com modelos de redes neurais rasas, as redes profundas têm a capacidade de identificar um número significativamente maior de classes.

O *Inception-V3*, por exemplo, opera com 1.000 neurônios na última camada, permitindo que ele reconheça um número correspondente de itens. Consequentemente, quando apresentado a uma nova amostra, o *Inception-V3* pode identificar e classificar com acurácia os objetos presentes nela.

Muitos programas antivírus capazes de detectar centenas de categorias de ame-

aças foram criados com base em modelos de DL que possuem múltiplos neurônios em sua camada de saída. Cada neurônio identifica o nível de ameaça de uma determinada entrada com base em uma classe existente (por exemplo, Citadel, Zeus, Ransomware, etc.). Há uma crença predominante de que, devido às poderosas capacidades dos modelos de deep learning, eles são capazes de alcançar maior acurácia em todas as categorizações de software malicioso.

Um antivírus de propósito geral baseado em *deep learning* pode se destacar na classificação de vários tipos de ameaças, mas ainda pode falhar na detecção de *malwares* específicos dentro de uma determinada categoria, em comparação com um antivírus especializado. Este último, projetado para um tipo específico de ameaça, tende a superar o antivírus de propósito geral, alcançando excelentes resultados, mas apenas na detecção de uma classe particular de *malware*.

É importante escolher métodos que combatam *malwares* direcionados com acurácia. Isso requer uma abordagem estratégica. Nesse contexto, a aplicação de *deep learning* pode não ser a solução mais viável. Para ilustrar essa questão, considere a rede neural profunda *Inception-V3*. Ela é complexa, com 23,6 milhões de parâmetros ajustáveis. Tais parâmetros exigem um grande conjunto de dados para um treinamento completo do modelo. Em termos simples, o *deep learning* tem fome de dados. Um grande número de parâmetros ajustáveis precisa de uma grande quantidade de dados de entrada. Na área de cibersegurança, devemos observar que dados rotulados de ameaças à IoT podem ser escassos.

O *Deep Learning* funciona melhor com muitos dados de treinamento. Isso permite que a rede aprenda a lidar com diferentes cenários. No entanto, se o modelo tiver apenas milhares, e não milhões, de amostras em seu conjunto de dados, isso pode comprometer sua capacidade de generalização.

Para um antivírus que combate ameaças à IoT, o *Deep Learning* pode parecer uma escolha tentadora. Mas devemos considerar alguns pontos. Imagine que o *Deep Learning* é como um carro de corrida potente que precisa de uma estrada ampla e desobstruída para mostrar todo o seu potencial. No entanto, se a estrada estiver

cheia de curvas fechadas e obstáculos, esse carro pode não ser a melhor opção. No contexto da cibersegurança, o *Deep Learning* é como esse carro de corrida. Ele requer grandes quantidades de dados para operar – ele devora dados. Se não tivermos uma grande quantidade de dados disponíveis, isso pode comprometer o desempenho do *Deep Learning*.

O *Deep Learning* é muito eficaz com muitos dados. Mas, para um antivírus focado em ameaças específicas, como *malwares* de IoT, outros métodos podem funcionar melhor. O aprendizado supervisionado com conjuntos de dados menores e focados pode ser mais eficiente. Essas abordagens podem ser uma maneira melhor de desenvolver soluções de cibersegurança. As organizações podem adaptá-las a ameaças específicas, como *malwares* de IoT.

Neste trabalho, defendemos o desenvolvimento de múltiplos programas antivírus especializados. O objetivo é atingir a máxima acurácia sem negligenciar categorias distintas de *malware*. Antivírus especializados também podem ser interpretados como uma forma de evitar o grande consumo de tempo de treinamento de modelos massivos de *deep learning*. Nesse cenário, múltiplas unidades de processamento são independentes e podem ser treinadas em paralelo, reduzindo ainda mais o tempo de treinamento da solução como um todo.

O tempo prolongado de treinamento representa uma desvantagem ao empregar redes de *deep learning*, e essa duração pode ser ainda maior porque as redes profundas são difíceis de treinar em paralelo devido à sua estrutura sequencial de camadas. Consequentemente, o cálculo de uma camada subsequente só pode começar após a conclusão da anterior.

Considerando que uma média de sete novas instâncias de *malware* é criada a cada segundo (Intel, 2018), é evidente que há uma alta demanda pelo retreinamento frequente dos modelos de redes neurais dos antivírus. Esse aspecto se torna um obstáculo significativo. Idealmente, o tempo necessário para treinar o software antivírus deve estar alinhado com o ritmo em que novas gerações de *malware* surgem globalmente.

Outro grande problema com as redes profundas é que elas possuem muitos parâmetros ajustáveis. Por exemplo, a rede profunda *Inception-V3* possui 23,6 milhões de parâmetros ajustáveis (CHOLLET, 2017). A otimização dos tempos de treinamento não seria possível mesmo para um supercomputador hipotético com milhões de processadores. O paradigma produtor-consumidor impede que todas as camadas funcionem simultaneamente, e o modelo sequencial em cascata das redes profundas representa um desafio significativo em termos de computação paralela. Como o nome sugere, uma camada só pode ser executada após a camada anterior ter concluído seu trabalho.

No entanto, existem alguns modelos de *deep learning* em estágio inicial capazes de processar dados simultaneamente. Atualmente, suas precisões podem não ser adequadas para uma ampla gama de aplicações (PINHEIRO et al., 2022). Os resultados de treinamento dessas redes ainda são estatisticamente inferiores aos de modelos sequenciais profundos e rasos (PINHEIRO et al., 2022). Portanto, não há evidências que sugiram que uma rede paralela profunda possa alcançar resultados satisfatórios quando aplicada a um antivírus.

Mesmo com um longo tempo de treinamento, os modelos de *deep learning* são capazes de alcançar altas taxas de acurácia. Explicar racionalmente essas taxas é fonte de muitas conjecturas e hipóteses, e também pode ser possível criar excelentes descritores de frequência durante o período de treinamento. Esses descritores estariam imersos nos milhões de parâmetros ajustáveis da arquitetura de DL. Identificá-los pode ser uma tarefa assustadora. A complexidade de identificação não reside apenas durante o treinamento dessas redes, mas também na identificação desses descritores e sua contribuição para a explicação do processo de decisão.

Em sua forma convencional, os modelos de DL frequentemente rotulam dados sem fornecer explicações detalhadas. Ferramentas contemporâneas frequentemente estabelecem padrões sem oferecer suporte significativo, o que pode solidificar a base de suas decisões. O desafio está na capacidade da rede de fornecer explicações que sejam compreensíveis para os seres humanos.

As redes profundas são modelos computacionais que utilizam grafos de profundidade. A estrutura é convencionalmente construída contendo múltiplas camadas sequenciais. Arquiteturas de redes profundas no estado da arte empregam camadas contendo diferentes tipos de processamento. Tipicamente, a diferenciação entre as camadas diz respeito a funções de ativação, normalização, convolução e redução de dimensionalidade [1].

Matematicamente, as redes neurais profundas, especialmente as redes convolucionais, são baseadas na convolução de filtros lineares. Embora desempenhe um papel fundamental em aplicações computacionais, esse filtro é limitado a aplicações em que um gradiente de fluxo vetorial é estabelecido.

Como exemplo, se observarmos cuidadosamente imagens biomédicas de máquinas de mamografia, é claro que as imagens estão repletas de ruído, o que dificulta a detecção de uma lesão na mama (LIMA et al., 2014). A convolução dos filtros é, portanto, crucial para a remoção do ruído. Dessa forma, ela descarta pequenas irregularidades no diagnóstico que correspondem a uma possível doença cancerígena. Para reduzir o ruído em imagens biomédicas (LIMA et al., 2014), técnicas de convolução, como filtros Gaussianos, são essenciais.

Como contraexemplo, considere o repositório mostrado na Tabela 3.1. Apesar de pertencerem à mesma vizinhança, os recursos estão separados uns dos outros. Um aplicativo suspeito que pode estar escaneando dados *WiFi* não está diretamente conectado ao acesso à galeria de imagens da vítima ou aos dados de navegação. Portanto, a convolução linear dos filtros consideraria tais operações como ruído, quando deveria ser o contrário, apenas porque a vizinhança tem valores positivos. Em síntese, o executável suspeito seria acusado de invadir o navegador da vítima, mesmo extraindo recursos que, de outra forma, não confeririam. Assim, as técnicas de convolução têm uma desvantagem quando aplicadas à detecção de padrões de *malware*.

[1] Exemplo de arquitetura de rede neural profunda. Disponível em: https://se.mathworks.com/help/deeplearning/gs/create-simple-image-classification-network-using-deep-network-designer.html. Acesso em maio de 2021.

3.3 A Proposta de Antivírus Especializados

Dada as limitações das abordagens gerais, a criação de antivírus especializados tem se mostrado uma solução promissora. Esses antivírus são projetados para detectar ameaças específicas, como *malwares* de IoT, com maior acurácia e eficiência. Neste trabalho, propomos um antivírus especializado baseado em redes neurais rasas, que combina alta acurácia com tempo de treinamento reduzido.

Para fundamentar nossa estrutura teórica, desenvolvemos um antivírus especializado capaz de detectar *malwares* de IoT. Isso serve como base para a criação de novos antivírus especializados em categorias específicas, utilizando nosso modelo personalizado de rede neural.

Nosso antivírus utiliza redes rasas em vez de redes convolucionais profundas. Como experimento, nosso antivírus pode combinar alta acurácia com tempo de aprendizado reduzido. Sua acurácia foi comparada à de um antivírus de última geração que utiliza redes neurais rasas e profundas.

Para evitar comparações injustas, a fase de extração de recursos é padronizada, monitorando 370 características que o arquivo suspeito pode realizar. O antivírus autoral leva frações de segundo para concluir seu treinamento, enquanto o antivírus baseado em *deep learning* leva horas.

Essa padronização durante a fase de extração é fundamental. Antivírus do estado da arte visam diferentes dispositivos, como *Android* e IoT, e também utilizam diferentes repositórios de aprendizado estatístico, o que deve ser levado em consideração ao analisar os dados da Tabela 3.1. Os valores apresentados nessa tabela foram replicados usando nosso conjunto de dados, o que nos permite comparar resultados. A Seção 6 mostra os resultados da réplica do antivírus do estado da arte e do nosso antivírus.

Tabela 3.1: Exemplo de um repositório de estatísticas baseado na detecção de atividades maliciosas.

Features		
Checagem dados de Wi-fi	Acesso ao Browser	Acesso a galeria de imagens
1	0	1

Capítulo 4

Metodologia

4.1 Métodos e materiais utilizados

Com relação ao material utilizado, este trabalho propõe um banco de dados que visa a classificação de executáveis benignos e *malwares* do tipo *Mediyes*. Existem 1.050 *malwares* e 1.050 outros executáveis benignos. Portanto, o conjunto de dados é adequado para aprender com inteligência artificial, já que ambas as classes de executáveis têm a mesma quantidade de executáveis.

Pragas virtuais foram extraídas de bancos de dados fornecidos por grupos de estudo entusiastas como o VirusShare, que é um repositório de amostras de *malware* para fornecer aos pesquisadores de segurança, analistas forenses e aos interessados em geral acesso a amostras de código malicioso. Quanto aos executáveis benignos, a aquisição veio de repositórios de aplicativos benignos como *SourceForge*, *GitHub* e *Sysinternals*. Dito isto, deve-se notar que todos os executáveis benignos foram submetidos ao VirusTotal e todos tiveram sua benignidade atestada pelos principais antivírus comerciais em todo o mundo. O diagnóstico, fornecido pelo VirusTotal, correspondente aos executáveis benignos e maliciosos está disponível no endereço virtual do banco de dados (Mediyes, 2024).

O objetivo da criação do banco de dados é dar total possibilidade à metodologia proposta de ser replicada por terceiros em trabalhos futuros. Assim, o trabalho realizado, disponibilizando gratuitamente a sua base de dados, permite a transpa-

rência e imparcialidade à pesquisa, além de demonstrar a veracidade dos resultados alcançados. Portanto, espera-se que a metodologia utilizada sirva de base para a criação de novos trabalhos científicos.

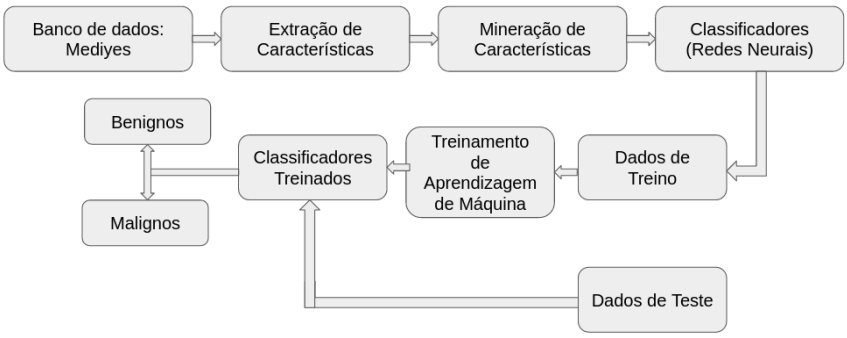
4.2 Metodologia proposta

Após identificar as limitações dos antivírus comerciais, o trabalho proposto visa criar um antivírus, dotado de inteligência artificial, capaz de diferenciar preventivamente aplicações de malware de benignas. A Figura 4.2 apresenta a metodologia proposta em um diagrama de blocos.

Quanto ao processamento de todos os dados, todos os experimentos foram realizados em um computador com 24 GB de RAM ("*random access memory*") e de 6 núcleos físicos (processadores) e uma GPU de 2048 núcleos CUDA e 4GB de memória.

O diagrama abaixo divide a metodologia em partes lógicas. Primeiramente, a parte de aquisição de dados a partir do banco de dados *Mediyes*. Após a extração desses recursos, é realizado o tratamento desses dados de forma que se tornem adequados para o processamento. Em sequência, é aplicada a classificação dos dados por meio da tecnologia proposta (Máquina de Aprendizagem Extrema) e, por último, é feita a validação cruzada com o intuito de garantir a eficácia do processo desenvolvido. O diagrama apresenta também como é realizado o processo de treinamento e testagem dos dados.

Figura 4.1: Diagrama da metodologia proposta.



Fonte: O autor (2025).

4.3 Extração de recursos

A extração de recursos de arquivos executáveis é realizada por meio de um processo de desmontagem (*disassembly*), que permite a decomposição do código binário em instruções de montagem. Esse procedimento é essencial para estudar os mecanismos presentes no executável de referência e, posteriormente, classificá-los por meio da rede neural descrita neste trabalho. Para a análise, foram extraídos 370 recursos de cada arquivo executável, utilizando ferramentas autorais e o *PEScanner*, uma ferramenta especializada na extração de informações de arquivos no formato PE (*Portable Executable*). Para facilitar a interpretação dos neurônios da camada de entrada da rede neural, o repositório utilizado amplia a descrição das propriedades auditadas para o antivírus (Mediyes, 2024).

- **Histograma de Instruções em Linguagem de Montagem**. O histograma de instruções fornece uma visão geral das operações realizadas pelo executável em nível de memória. Essa análise permite identificar padrões de comportamento, como o uso excessivo de operações de leitura/escrita ou chamadas de sistema, que podem indicar atividades suspeitas.

- **Sub-rotinas que invocam TLS (*Transport Layer Security*)**. A presença de sub-rotinas que utilizam TLS pode indicar comunicação criptografada, comum em *malwares* que tentam ocultar suas atividades.

- **Sub-rotinas de exportação de dados**. Sub-rotinas responsáveis por exportar dados podem sugerir que o executável está tentando extrair informações sensíveis do sistema.

- **APIs (*Application Programming Interface*) Utilizadas**. A análise das APIs usadas pelo executável é fundamental para identificar comportamentos maliciosos. Por exemplo, APIs relacionadas à manipulação de arquivos, rede ou registro do Windows podem indicar atividades como coleta de dados, persistência no sistema ou comunicação com servidores remotos.

- **Indícios de Fragmentação de Disco e Inicialização Inválida**. Recursos relacionados à fragmentação do disco rígido e tentativas de inicialização inválida podem sugerir que o *malware* está tentando corromper o sistema de arquivos ou dificultar a análise forense.

- **Modo de Execução do Aplicativo**. O modo de execução do aplicativo pode ser dividido em duas categorias:

  - Software com Interface Gráfica (GUI): Utiliza janelas interativas para interação com o usuário.

  - Software em Execução no Console: Opera diretamente em terminais de comando, sem interfaces visuais elaboradas.

- **Recursos Relacionados ao Sistema Operacional**. A análise forense investiga informações internas do Windows, como *drivers* e configurações do sistema. Esses recursos podem revelar técnicas de persistência ou manipulação do sistema operacional por parte do *malware*.

- **Registro do Windows (*Regedit*)**. O registro do Windows é um alvo comum para *malwares* que desejam persistir no sistema. A inserção de entradas maliciosas no *Regedit* permite que o *malware* seja reiniciado automaticamente com o sistema, mesmo após sua detecção e remoção aparente. Por exemplo,

um *malware* pode redirecionar a página inicial do *Internet Explorer* para um site malicioso, reiniciando o ataque a cada inicialização.

- **Funcionalidades Relacionadas a *Spywares***.
  - *Keyloggers*: Capturam informações do teclado para roubo de senhas e logins.
  - *Screenloggers*: Capturam imagens da tela da vítima, permitindo o monitoramento de atividades sensíveis.
  - O antivírus monitora se o arquivo suspeito tenta coletar informações privadas do usuário, como credenciais de acesso ou dados bancários.
- **Técnicas de Anti-Forense Digital**. Essas técnicas incluem a remoção, ocultação ou subversão de evidências, com o objetivo de dificultar a análise forense. Por exemplo:
  - Suspensão temporária da execução do *malware* para evitar detecção.
  - Desativação de ferramentas de segurança, como *firewalls* e antivírus.
  - Ocultação de arquivos infectados por meio de esteganografia.
- **Criação de GUI (*Graphical User Interface*)**. A criação de interfaces gráficas por parte do *malware* pode ser usada para enganar o usuário, simulando aplicativos legítimos enquanto realiza atividades maliciosas em segundo plano.
- **Uso Ilícito da RAM**. O antivírus investiga se o aplicativo suspeito tenta reservar, confirmar ou alterar o estado de uma região da memória RAM. Esse comportamento pode indicar tentativas de injetar código malicioso em processos legítimos.
- ***Sniffers* de Rede**. *Sniffers* são utilizados para capturar dados de pacotes de rede. O antivírus verifica se o aplicativo suspeito tenta ler informações sensíveis, como credenciais de login ou dados transmitidos em texto claro.

- **Soquetes**. Em aplicações convencionais, soquetes são criados no servidor para aguardar conexões de usuários. No entanto, *malwares* podem criar soquetes no sistema local, aguardando conexões de computadores remotos para extrair informações sensíveis, como senhas ou fotos.

- **Tráfego de Rede**. A análise de tráfego de rede verifica se o arquivo suspeito tenta consultar servidores DNS ou criar sessões FTP/HTTP em tempo de execução. Essas atividades podem indicar comunicação com servidores de comando e controle (C2) ou tentativas de coleta de dados.

- **Programas Utilitários**. Recursos relacionados a programas utilitários podem incluir ferramentas de compressão, criptografia ou manipulação de arquivos, frequentemente utilizadas por *malwares* para ocultar suas atividades.

É importante destacar que, individualmente, nenhum desses recursos representa necessariamente um comportamento malicioso. Por exemplo, o uso de TLS ou a criação de soquetes pode ser legítimo em aplicações benignas. Portanto, a detecção de *malware* deve ocorrer por meio do cruzamento de informações e da ponderação de todos os recursos analisados. A combinação de múltiplos indicadores suspeitos aumenta a confiabilidade da detecção, permitindo uma análise mais precisa e robusta.

4.4 Classificação

Quanto ao reconhecimento de padrões de *malware*, a tarefa principal é relacionada a atribuir a cada arquivo investigado uma classe (rótulo) com base em suas características. Então, com base em um conjunto de arquivos chamado conjunto de treinamento, podem ser feitas hipóteses sobre as diferentes classes associadas ao antivírus proposto. A partir disso, o classificador estima a classe de documentos inéditos comparando características comportamentais auditadas a tempo com aquelas capturadas durante a fase de treinamento.

O objetivo do classificador é obter uma função de separação entre as classes do antivírus (*malware*, benigno). Desta forma, quando é apresentado um executável inédito, a função é aplicada e, em seguida, atribui uma classe à qual este arquivo deve pertencer. Matematicamente, $c = f(x)$, onde $x = x_1, x_2, \ldots, x_t$ é o vetor traçado a partir do arquivo investigado, t corresponde à quantidade de recursos de entrada analisados e c à classe. Finalmente, f é a função do mapeamento do classificador.

Ao estabelecer um classificador linear, o classificador representa uma linha que possui a função de separar os padrões de classes diferentes. Portanto, cada caso investigado será classificado de acordo com o lado da linha em que está mapeado. Visando o reconhecimento comportamental dos *malwares* modernos, devem ser usados classificadores que possam construir uma separação não linear entre as classes. Para comprovar o embasamento teórico, o antivírus autoral emprega redes neurais não lineares, especificamente, redes neurais morfológicas extremas.

Capítulo 5

Resultados

5.1 Resultados das redes ELM

Sete tipos diferentes de *kernel* foram usados nas redes neurais. No estado da arte, cinco desses *kernels* são descritos por HUANG et al. (2012) e são eles: *Wavelet*, Sigmoide, Seno, *Hard Limit* e *Tribas Transforms* (funções de base trigonométricas) (HUANG, 2012). Além disso, *kernels* autorais adicionais são empregados: Pseudo-dilatação e Pseudo-erosão.

O *kernel* do tipo *Wavelets* não possui camada oculta (HUANG, 2012). Os cálculos são baseados na transformação dos dados de entrada e podem funcionar de forma similar aos *kernels* que contêm arquiteturas com camadas ocultas (HUANG, 2012). Uma boa capacidade de generalização desses *kernels* depende de uma escolha ajustada de parâmetros (C, γ) (HUANG, 2012). O parâmetro de custo C refere-se a um ponto de equilíbrio razoável entre a largura da margem do hiperplano e a minimização do erro de classificação em relação ao treinamento. Já o parâmetro γ controla o limite de decisão em função das classes (HUANG, 2012). Não existe um método universal com relação ao sentido de escolha dos parâmetros (C, γ).

Neste trabalho, há a investigação dos parâmetros (C, γ) inspirado no método proposto por (HUANG, 2012), que consiste em treinar sequências crescentes de C e γ, matematicamente, 2^n, onde $n = \left\{-24, -10, 0, 10, 25\right\}$ (HUANG, 2012). A hipótese é verificar se esses parâmetros com valores diferentes dos padrões, ($C =$

$1, \gamma = 1$), geram melhores resultados. No núcleo linear, há apenas a investigação do parâmetro de custo C, pois não é possível explorar o parâmetro γ (HUANG, 2012).

Para cada combinação utilizada, é empregada validação cruzada através do método *k-fold*, onde $k = 10$. O objetivo da utilização desse método é que os resultados alcançados não sejam influenciados por conjuntos de treinamento e teste. Por isso, o total de dados é dividido em dez partes.

Na primeira execução, a primeira parte é destinada ao conjunto de testes, enquanto a outra é reservada para o treinamento. Essa alternância ocorre por dez execuções até que todas as partes tenham sido aplicadas à fase de teste. A acurácia do ELM é dada pela média aritmética da taxa de acerto obtida nas dez iterações.

Como mencionado anteriormente, na rede ELM não há retropropagação de dados. Portanto, o objetivo do método de validação cruzada *k-fold* não é estabelecer uma parada de critério para evitar *overfitting* (excesso de treinamento), mas para verificar que o classificador sofre mudanças abruptas em sua acurácia dependendo dos conjuntos destinados a treinamento e teste. Dito isto, o objetivo é que classificadores tendenciosos, em relação a uma determinada classe, não tenham suas taxas de acurácia favorecidas.

A Tabela 5.1 detalha os resultados obtidos pelas redes neurais ELMs com *kernel Wavelets*. Cada linha nesta tabela contém 10 execuções referentes à validação cruzada do método *k-fold*, onde $k = 10$. Em relação à acurácia na fase de teste, o desempenho médio máximo foi de 76,24% na distinção entre casos benignos e malignos com os parâmetros $(C, \gamma) = (2^{-10}, 2^{-24})$. Na Tabela 5.1, há apenas as descrições de melhor e pior caso, nesta ordem, para cada núcleo.

A Tabela 5.2 apresenta os resultados alcançados pela rede ELM trabalhando com *kernel* linear. É feita a análise apenas do parâmetro C, pois não é possível explorar o parâmetro γ em um *kernel* linear (HUANG, 2012). Cada linha na Tabela 5.2 contém 10 execuções distintas sobre o *k-fold* na validação cruzada, onde $k = 10$. O máximo e mínimo de acurácia foram de 93,48% e 50,00%, respectivamente. Visto isso, é possível afirmar que a investigação do parâmetro de custo C é capaz de maximizar

a acurácia na identificação.

A Tabela 5.3 mostra os resultados obtidos pela rede ELM utilizando os *kernels Sigmoid*, Sine, Hard Limit, Tribas (funções de base trigonométricas), pseudo-dilatação e pseudo-erosão. Eles empregam uma arquitetura de camada oculta. Em seguida, é apresentada a análise em relação à quantidade de neurônios na camada oculta desses núcleos. A hipótese é verificar se as arquiteturas que exigem maior volume de cálculos, como dobrar o número de neurônios na camada oculta, são capazes de gerar melhores taxas de acurácia em comparação com arquiteturas que exigem uma quantidade menor de cálculos. Partindo disso, é feita a avaliação de dois tipos de arquiteturas: são empregados 100 e 500 neurônios em suas respectivas camadas ocultas. Estas arquiteturas possuem antecedentes de excelente acurácia na aplicação de redes ELM na área de Engenharia Biomédica (LIMA et al., 2021). Cada linha da Tabela 5.3 contém 10 execuções distintas referentes ao método k-fold, onde $k = 10$. Em relação à acurácia, o desempenho médio máximo foi de 99,25% com desvio padrão de 0,45% através do *kernel* de pseudo-dilatação dotado de 500 neurônios em sua camada oculta.

Tabela 5.1: Resultados das redes neurais ELM. Os parâmetros (C, γ) variam de acordo com o conjunto $\{2^{-24}, 2^{-10}, 2^{0}, 2^{10}, 2^{25}\}$. São exibidas apenas as melhores e piores acurácias.

kernel	(C, γ)	Acurácia de treinamento (%)	Acurácia de teste (%)	Tempo de treino (seg.)	Tempo de teste (seg.)
Polinomial	$(2^{-10}, 2^{0})$	**98,63 ± 0,09**	**98,55 ± 0,46**	1,09 ± 0,08	0,04 ± 0,01
	$(2^{-10}, 2^{25})$	50,01 ± 0,00	49,87 ± 0,00	1,82 ± 0,08	0,12 ± 0,01
Wavelets	$(2^{10}, 2^{-24})$	100,00 ± 0,00	97,40 ± 0,82	1,56 ± 0,06	0,10 ± 0,01
	$(2^{-10}, 2^{10})$	57,02 ± 0,61	54,42 ± 2,49	1,68 ± 0,05	0,11 ± 0,01

Fonte: O autor (2025).

Tabela 5.2: Resultados das redes neurais ELM com o *kernel* Linear. Os parâmetros C variam de acordo com o conjunto $\{2^{-24}, 2^{-10}, 2^{0}, 2^{10}, 2^{25}\}$. São exibidas apenas as melhores e piores acurácias.

kernel	C	Acurácia de treinamento (%)	Acurácia de teste (%)	Tempo de treino (seg.)	Tempo de teste (seg.)
Linear	2^{-10}	**98,53 ± 0,05**	**98,45 ± 0,50**	1,04 ± 0,05	0,03 ± 0,00
	2^{-24}	50,00 ± 0,02	50,00 ± 0,14	1,02 ± 0,06	0,03 ± 0,01

Fonte: O autor (2025).

Tabela 5.3: Resultados das redes ELM. O número de neurônios na camada oculta varia de acordo com os dados 100, 500.

kernel	neurônios	Acurácia de treinamento (%)	Acurácia de teste (%)	Tempo de treino (seg,)	Tempo de teste (seg,)
Sigmoide	500	96,71 ± 0,57	91,96 ± 1,70	0,72 ± 0,03	0,01 ± 0,00
	100	80,44 ± 0,26	79,97 ± 1,87	0,33 ± 0,02	0,00 ± 0,00
Seno	500	90,01 ± 0,27	81,26 ± 2,13	0,74 ± 0,03	0,01 ± 0,00
	100	79,89 ± 0,33	77,53 ± 2,90	0,34 ± 0,02	0,00 ± 0,00
Hard limit	100	50,01 ± 0,00	49,87 ± 0,00	0,33 ± 0,02	0,00 ± 0,00
	500	50,01 ± 0,00	49,87 ± 0,00	0,70 ± 0,02	0,00 ± 0,01
Tribas	100	50,00 ± 0,02	50,00 ± 0,14	0,32 ± 0,01	0,00 ± 0,00
	500	50,00 ± 0,02	50,00 ± 0,14	0,57 ± 0,03	0,01 ± 0,00
Dilatação Clássica	500	**99,71 ± 0,04**	**99,25 ± 0,45**	2,66 ± 0,07	0,20 ± 0,01
	100	99,17 ± 0,05	99,01 ± 0,28	0,81 ± 0,04	0,05 ± 0,01
Erosão Clássica	500	96,77 ± 0,53	95,82 ± 1,31	2,71 ± 0,05	0,22 ± 0,01
	100	82,94 ± 0,65	82,20 ± 3,03	0,83 ± 0,04	0,05 ± 0,01

Fonte: O autor (2025).

5.2 Resultados em relação ao estado da arte

Nesta seção, o antivírus autoral é comparado com os antivírus atuais. Para evitar comparações injustas, o estágio de extração de recursos é padronizado monitorando 370 comportamentos que o executável suspeito pode fazer quando executado propositalmente. O antivírus autoral aplica redes morfológicas superficiais utilizando um *kernel* de pseudo-dilatação, além disso, sua camada oculta possui 500 neurônios.

Por outro lado, o antivírus por (LIMA et al., 2021) emprega redes neurais superficiais baseadas em retropropagação, onde onze funções distintas de aprendizado são investigadas para otimizar a acurácia de seu antivírus. Para cada função de aprendizagem, (LIMA et al., 2021) explora 4 arquiteturas de camadas ocultas.

O antivírus proposto também é comparado aos antivírus baseados no conceito de rede neural profunda. No presente trabalho, são replicados os antivírus feitos por (SU and VASCONCELLOS, 2018), (HOU and SAAS, 2016), (MANIATH and ASHOK, 2017), (HARDY and LINGWEI, 2016) e (FARUKI and BUDDHADEV, 2019). Além disso, o *firewall* desenvolvido por (WOZNIAK and SILKA, 2015) foi avaliado. O trabalho foi replicado utilizando o conjunto de dados próprio para evitar comparações injustas.

A Figura 5.1 e a Figura 5.2 são representações gráficas dos resultados descritos na Tabela 5.4. A Figura 5.1 (a) mostra os *boxplots* para a melhor acurácia no treinamento. O antivírus autoral obteve uma acurácia média de 99,71%. O antivírus feito por (LIMA et al., 2021) obteve acurácia média de 48,84% e 99,90%, em seus piores e melhores cenários, respectivamente. Esses resultados foram obtidos usando as funções de aprendizado *"Resilient backpropagation"* e *"Conjugate gradient backpropagation with Fletcher-Reeves updates"*, respectivamente. A pior arquitetura tem uma única camada oculta contendo 500 neurônios, enquanto a melhor arquitetura possui duas camadas contendo 100 neurônios em cada uma.

A Figura 5.1 (b) apresenta os boxplots, na fase de teste, em relação aos antivírus autoral e de última geração. O antivírus autoral obteve desempenho médio de 99,25% com desvio padrão de 0,45%. O antivírus feito por (LIMA et al., 2021) obteve acurácia média de 48,84% e 99,83%, em seus piores e melhores cenários, respectivamente. Partindo desse fator, corrobora-se que as redes neurais baseadas em retropropagação podem sofrer grandes variações, em suas acurácias, dependendo de seus parâmetros de configuração. Então, a decisão tomada por (LIMA et al., 2021) foi sensata. Este antivírus de última geração explora diferentes funções de aprendizado, gradientes e arquiteturas para otimizar a acurácia de suas redes neurais com base na retropropagação de dados.

A Figura 5.2 (a) e a Figura 5.2 (b) mostram boxplots envolvendo o tempo gasto nas fases de treinamento e teste, respectivamente. O antivírus autoral consome apenas 2,66 segundos para concluir, em média, seu treinamento. Em relação ao tempo de treinamento, o antivírus feito por (HOU and SAAS, 2016) é o mais lento, consumindo 4.040,26 segundos. Em relação ao tempo gasto durante a fase de teste, todas as técnicas consumiram tempos muito próximos, sem grandes discrepâncias.

A Tabela 5.5 mostra as matrizes de confusão das técnicas apresentadas na Tabela 5.4 em termos percentuais. A matriz de confusão é importante para verificar a qualidade do aprendizado supervisionado. Na Tabela 5.5, B. e M. são abreviações de Benigno e *Malware*. As classes desejadas estão dispostas na etiqueta vertical,

enquanto as classes obtidas estão na etiqueta horizontal. Na matriz de confusão, a diagonal principal é ocupada por casos em que a classe obtida coincide com a classe esperada, denominados casos verdadeiros positivos. Logo, um bom classificador tem a diagonal principal ocupada por valores altos e outros elementos possuem valores baixos.

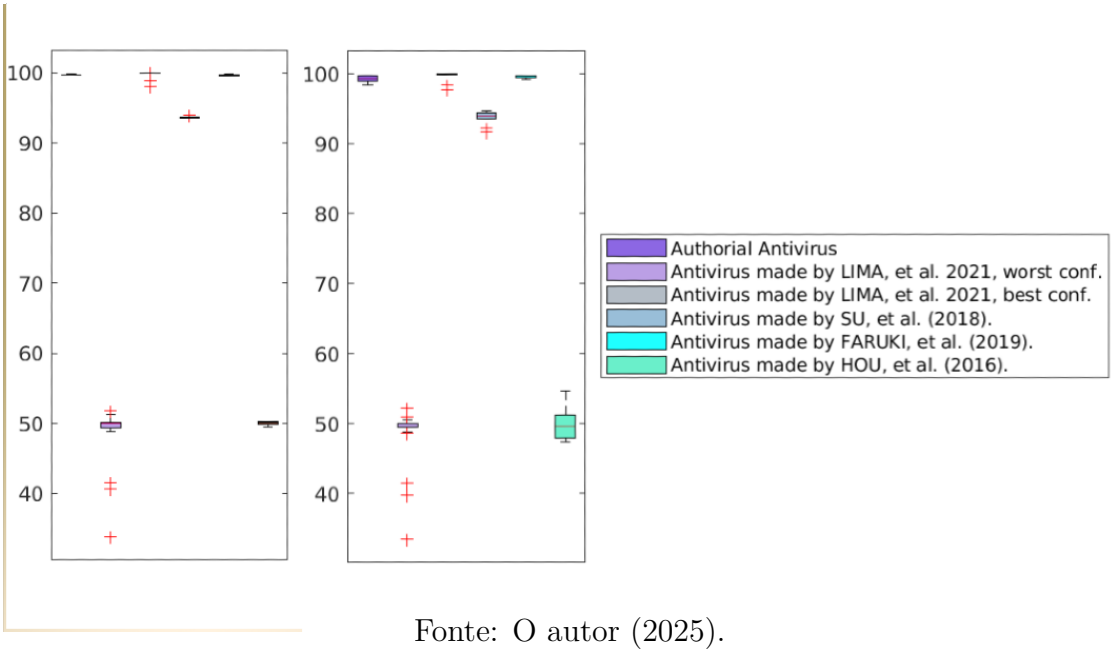
A Tabela 5.5 mostra as principais diagonais destacadas em negrito. O antivírus proposto, em fase de teste, classificou erroneamente em média 2,84% dos casos como benignos quando eram casos de *malware* (falso negativo). Seguindo o mesmo raciocínio, houve uma classificação média de 0,51% dos casos ditos *malware* quando eram na verdade aplicativos benignos (falso positivo).

Ainda em relação à Tabela 5.5, a sensibilidade e a especificidade referem-se à capacidade do antivírus em identificar *malware* e identificar aplicativos benignos, respectivamente. O trabalho proposto apresenta a matriz de confusão em termos percentuais para facilitar a interpretação da sensibilidade e especificidade. Em síntese, a sensibilidade e especificidade são apresentadas na própria matriz de confusão, descrita na Tabela 5.5. Por exemplo, o antivírus autoral tem uma média de 92,48% em relação tanto à sensibilidade quanto aos verdadeiros positivos. Seguindo o mesmo raciocínio, o antivírus autoral obtém, em média, 94,76% para ambas especificidades e verdadeiros negativos.

A Tabela 5.6 mostra os valores de *t-student* paramétricos e não paramétricos dos testes de hipótese de *Wilcoxon* entre o antivírus proposto e o estado da arte. É possível concluir que o antivírus autoral é estatisticamente diferente de todas as outras amostras, com exceção do antivírus feito por SU, *et al.* (2018). A explicação é que tanto no teste *t-student* paramétrico quanto no teste não paramétrico de *Wilcoxon*, a hipótese nula foi rejeitada.

O antivírus autoral demonstrou uma grande vantagem quando comparado ao estado da arte. O antivírus atinge um desempenho médio de 99,25% dentro de um treinamento médio de 2,66 segundos. Sabendo que 8 (oito) novos *malwares* são lançados a cada segundo (Intel, 2018), é logicamente coeso que um antivírus

Figura 5.1: Boxplots referentes às acurácias do antivírus autoral e do estado da arte.



Fonte: O autor (2025).

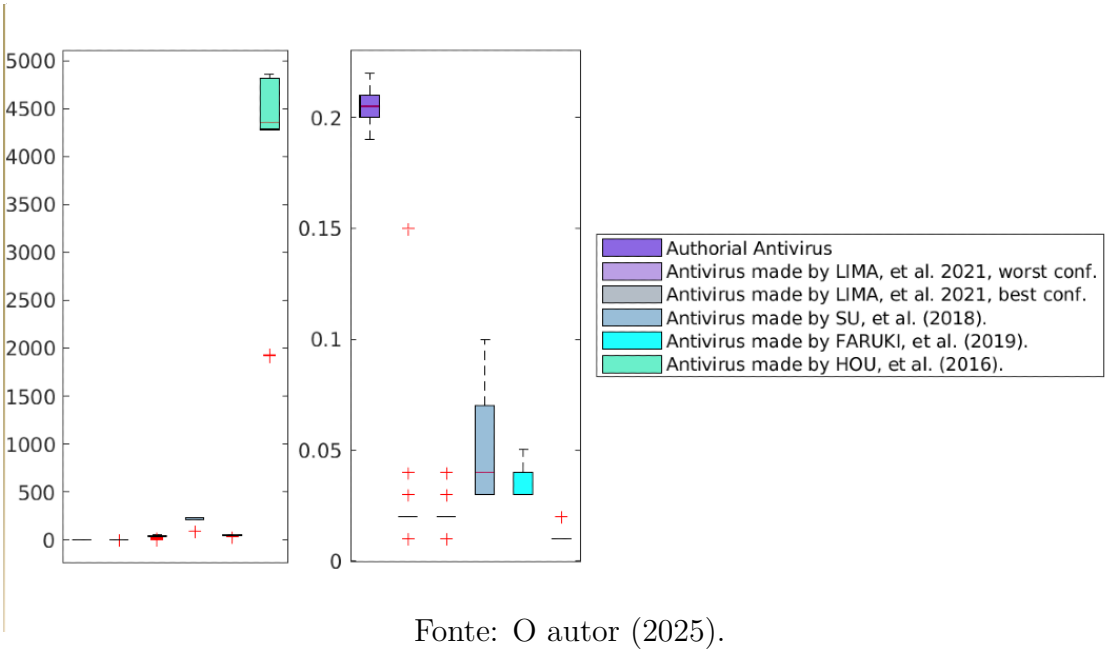
recém-lançado pode já estar obsoleto e exigir novo treinamento por meio de uma vulnerabilidade recém-descoberta. Em síntese, o tempo de aprendizado de um antivírus não deve ser discrepante em relação à taxa de criação de novos *malwares* em todo o mundo.

Tabela 5.4: Comparação entre o antivírus autoral e o estado da arte.

Técnica	Acurácia de treino (%)	Acurácia de teste (%)	Tempo de treino (seg.)	Tempo de teste (seg.)
Antivírus autoral	99,71 ± 0,04	99,25 ± 0,45	2,66 ± 0,07	0,20 ± 0,01
LIMA, *et al,* (2021), pior conf.	48,84 ± 3,67	48,84 ± 3,82	1,27 ± 0,93	0,02 ± 0,02
LIMA, *et al,* (2021), melhor conf.	99,90 ± 0,41	99,83 ± 0,50	38,65 ± 12,48	0,02 ± 0,01
SU, R, *et al,* (2018)	93,66 ± 0,11	93,65 ± 0,96	196,81 ± 54,84	0,05 ± 0,03
FARUKI, *et al,* (2019)	99,67 ± 0,08	99,52 ± 0,17	45,62 ± 6,10	0,04 ± 0,01
HOU, *et al,* (2016)	50,00 ± 0,26	50,00 ± 2,37	4040,26 ± 1139,70	0,01 ± 0,00

Fonte: O autor (2025).

Figura 5.2: Boxplots dos tempos de processamento do antivírus autoral e do estado da arte.



Fonte: O autor (2025).

Tabela 5.5: Matriz de confusão do Antivírus Autoral e Estado da Arte em (%).

		Treino		Teste	
Técnica		M.	B.	M.	B.
Antivírus autoral	M.	**99,42 ± 0,07**	0,00 ± 0,00	**98,68 ± 0,71**	0,16 ± 0,26
	B.	0,58 ± 0,7	**100,00 ± 0,00**	1,32 ± 0,71	**99,84 ± 0,26**
Antivírus de LIMA,	M,	**61,87 ± 48,15**	64,19 ± 46,53	**61,90 ± 48,22**	64,27 ± 46,70
et al. pior conf. (2021)	B,	38,13 ± 48,15	**35,81 ± 46,53**	38,10 ± 48,22	**35,73 ± 46,70**
Antivírus de LIMA,	M,	**99,85 ± 0,58**	0,06 ± 0,25	**99,76 ± 0,87**	0,10 ± 0,19
et al. melhor conf. (2021)	B,	0,15 ± 0,58	**99,94 ± 0,25**	0,24 ± 0,87	**99,90 ± 0,19**
Antivírus de SU,	M,	**98,20 ± 0,14**	10,10 ± 0,18	**98,17 ± 1,04**	10,13 ± 1,65
et al. (2018)	B,	1,80 ± 0,14	**89,90 ± 0,18**	1,83 ± 1,04	**89,87 ± 1,65**
Antivírus de	M,	**99,76 ± 0,15**	0,41 ± 0,07	**99,57 ± 0,43**	0,53 ± 0,51
FARUKI, *et al.* (2019)	B,	0,24 ± 0,15	**99,59 ± 0,07**	0,43 ± 0,43	**99,47 ± 0,51**
Antivírus de	M,	**0,00 ± 0,00**	0,00 ± 0,00	**0,00 ± 0,00**	0,00 ± 0,00
HOU, *et al.* (2016)	B,	100,00 ± 0,00	**100,00 ± 0,00**	100,00 ± 0,00	**100,00 ± 0,00**

Fonte: O autor (2025).

Tabela 5.6: T-students e Wilcoxon testam as hipóteses do antivírus autoral e do estado da arte.

	t-students (teste paramétrico)		Wilcoxon (teste não-paramétrico)	
Comparação	Hipóteses	*p*-value	Hipóteses .	*p*-valor
Antivírus autoral *vs* Antivírus de LIMA, *et al.* (2021), pior conf.	1	2,6677e-38	1	1,94438e-11
Antivírus autoral *vs* Antivírus de LIMA, *et al.* (2021), melhor conf.	1	5,38426e-05	1	4,18126e-06
Antivírus autoral *vs* Antivírus de SU, *et al.* (2018)	1	4,10589e-31	1	2,33755e-11
Antivírus autoral *vs* Antivírus de FARUKI, *et al.* (2019)	1	3,13276e-07	1	2,59838e-05
Antivírus autoral *vs* Antivírus de MANIATH, *et al.* (2017)	1	5,52345e-10	1	1,61456e-09
Antivírus autoral *vs* Firewall de WOZNIAK, *et al.* (2015)	1	1,9662e-38	1	2,33755e-11
Antivírus autoral *vs* Antivírus de HOU, *et al.* (2016)	1	2,07471e-17	1	2,37833e-11
Antivírus autoral *vs* Antivírus de HARDY, *et al.* (2016)	1	0,000682081	1	0,00702735

Fonte: O autor (2025).

Capítulo 6

Conclusão

Dado o crescente número de novos *malwares*, é de vital importância que as plataformas de detecção disponibilizem mecanismos de vigilância cibernética que atendam às demandas dos clientes de forma preventiva. Caso contrário, nos cenários em que ocorrem falhas na identificação de aplicativos maliciosos, há iminência de que dados confidenciais de clientes sejam disponibilizados por pessoas não autorizadas.

Assim, infere-se que a seleção do antivírus tem um papel importante no combate às pragas virtuais. Na avaliação apresentada, a variação na detecção de *malware* do tipo *Mediyes* foi de 0% a 99,52%, dependendo de qual antivírus comercial foi escolhido. O presente trabalho realizou a análise de 89 antivírus disponíveis comercialmente. Em média, eles conseguiram detectar 68,30% dos *malwares*. Feita a análise das amostras, foi possível identificar que os antivírus, em média, relataram falsos negativos e foram omitidos em 17,76% e 31,94% dos casos, respectivamente. No presente trabalho, a plataforma VirusTotal foi utilizada para submeter, de forma automatizada, o *malware* aos antivírus. Deve-se ressaltar que no VirusTotal, não existe a possibilidade de escolher a versão *shareware* dos antivírus. Então, não foi possível fazer comparações entre os recursos gratuitos e comerciais de um mesmo antivírus. Deduz-se que os serviços oferecidos nas versões *shareware* apresentam desempenho significativamente inferior ao das versões completas.

Vale ressaltar também que na análise apresentada, os *malwares* analisados são de domínio público, empregados em atividades maliciosas. Mesmo assim, uma grande

quantidade dos antivírus comerciais avaliados não tinha conhecimento sobre a existência dos arquivos infectados investigados.

Para suprir as limitações dos antivírus comerciais, foram desenvolvidos antivírus baseados em inteligência artificial que são capazes de auditar milhares de *malwares* e aprender, estatisticamente, quais são suas características maliciosas. Portanto, após o aprendizado, o antivírus inteligente pode identificar e classificar o malware recém-criado de acordo com a comparação entre seus recursos e os catalogados durante a fase de aprendizado. Pode-se tornar autônomo o aprendizado do comportamento do *malware*. Assim, não haveria necessidade de esperar um usuário ser infectado e, posteriormente, denunciar uma atitude suspeita, para só então o antivírus tomar alguma ação em relação à descoberta do *malware*.

Portanto, com o intuito de contribuir no combate à disseminação de arquivos maliciosos, foi desenvolvido um antivírus autoral capaz de classificar os executáveis entre benignos e *malware*. Ao todo, o antivírus monitora e pondera, estatisticamente, 370 ações que o arquivo suspeito pode realizar quando executado no sistema operacional. Em ambiente controlado, o antivírus monitora alterações no Registro (Banco de Dados) do sistema operacional e os rastreamentos de chamadas executadas por todos os processos gerados pelo *malware*. Todo o reconhecimento do padrão, referente às 370 ações suspeitas, é realizado por redes neurais extremas.

Ao invés de *kernels* convencionais, os *kernels* autorais são empregados para ELMs. A rede ELM foi escolhida pois tem como principal característica a velocidade de treinamento e previsão de dados assertiva quando comparada às redes neurais convencionais. O *kernel pseudo-Dilation* autoral é capaz de distinguir *malware* do tipo *Mediyes* de aplicativos benignos em 99,25% dos casos, acompanhado por um tempo de treinamento de 0,55 segundos.

Por último, é interessante destacar que o intuito deste trabalho é trazer uma visão diferente acerca do desempenho dos antivírus atuais, oferecendo alternativas criativas e eficientes de solucionar o problema de detecção de *malwares* do tipo *Mediyes*.

Referências

W. W. AZEVEDO and *et al.* Morphological extreme learning machines applied to detect and classify masses in mammograms. *In: 2015 International Joint Conference on Neural Networks (IJCNN), Killarney.*, 2015. doi: https://doi.org/10.1109/IJCNN.2015.7280774.

W. W. AZEVEDO and *et al.* Morphological extreme learning machines applied to the detection and classification of mammary lesions. *In: Tapan K Gandhi; Siddhartha Bhattacharyya; Sourav De; Debanjan Konar; Sandip Dey. (Org.). Advanced Machine Vision Paradigms for Medical Image Analysis. 1ed.Londres: Elsevier Science.*, pages 1–30, 2020. doi: https://doi.org/10.1016/B978-0-12-819295-5.00003-2.

François CHOLLET. Xception: Deep learning with depthwise separable convolutions. *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017. doi: 10.1109/CVPR.2017.195.

P. FARUKI and B. *et al.* BUDDHADEV. Droiddivesdeep: Android malware classification via low level monitorable features with deep neural networks. *International Conference on Security Privacy*, 2019. doi: https://doi.org/10.1007/978-981-13-7561-3_10.

W. HARDY and C. *et al.* LINGWEI. Dl 4 md : A deep learning framework for intelligent malware detection. *In Int'l Conf. Data Mining*, pages 61–67, 2016.

S. HOU and A. *et al.* SAAS. Droiddelver: An android malware detection system using deep belief network based on api call blocks. *Web-Age Information Management. WAIM 2016 International Workshops, MWDA, SDMMW, and SemiBDMA*, 2016. doi: https://doi.org/10.1007/978-3-319-47121-1_5.

G. B. *et al.* HUANG. Classification ability of single hidden layer feedforward neural networks. *The IEEE Transactions on Neural Networks and Learning Systems*, 11(3):799–801, 2000. doi: https://doi.org/10.1109/72.846750.

G. B. *et al.* HUANG. Extreme learning machine for regression and multiclass classification. *IEEE Transactions on Systems, Man, and Cybernetics*, 42(2): 513–519, 2012. doi: https://doi.org/10.1109/TSMCB.2011.2168604.

Intel. *McAfee Labs*. Accessed on Feb 2020, 2018. URL `https://www.mcafee.com/enterprise/en-us/assets/reports/rp-quarterly-threats-mar-2018.pdf`.

S. M. L. LIMA, A. G. SILVA-FILHO, and W. P. DOS SANTOS. A methodology for classification of lesions in mammographies using zernike moments, elm and svm neural networks in a multi-kernel approach. *In: 2014 IEEE International Conference on Systems, Man and Cybernetics SMC, San Diego*, 2014. doi: https://doi.org/10.1109/SMC.2014.6974041.

S. M. L. LIMA, SILVA-FILHO, and W. P. SANTOS. *Morphological Decomposition to Detect and Classify Lesions in Mammograms.In: Wellington Pinheiro dos Santos; Maíra Araújo de Santana; Washington Wagner Azevedo da Silva. (Org.). Understanding a Cancer Diagnosis*. 2020. URL `https://novapublishers.com/shop/understanding-a-cancer-diagnosis/`.

S.M.L LIMA. *Limitation of COTS antiviruses: issues, controversies, and problems of COTS antiviruses*. In: Cruz-Cunha, M.M., Mateus-Coelho, N.R. (eds.) Handbook of Research on Cyber Crime and Information Privacy, vol. 1, 1st edn. IGI Global, Hershey, 2020. doi: http://dx.doi.org/10.4018/978-1-7998-5728-0.ch020.

S.M.L. LIMA, A. G. SILVA-FILHO, and W. P. SANTOS. Detection and classification of masses in mammographic images in a multi-kernel approach. *Computer Methods and Programs in Biomedicine*, 134:11–29, 2016. doi: https://doi.org/10.1016/j.cmpb.2016.04.029.

S.M.L. LIMA, H.K.L. SILVA, and J.H.S. *et al.* LUZ. Artificial intelligence-based antivirus in order to detect malware preventively. *Progress in Artificial Intelligence*, 2021. doi: https://doi.org/10.1007/s13748-020-00220-4.

S. MANIATH and A. *et al.* ASHOK. Deep learning lstm based ransomware detection. *Recent Developments in Control, Automation Power Engineering*, 2017. doi: https://doi.org/10.1109/RDCAPE.2017.8358312.

Mediyes. Retrieval for mediyes malware analysis. *O Autor*, Disponível em: `https://github.com/DejavuForensics/Mediyes`. 2024.

J. M. S. *et al.* PEREIRA. *Method for Classification of Breast Lesions in Thermographic Images Using ELM Classifiers. In: Wellington Pinheiro dos Santos; Maíra Araújo de Santana; Washington Wagner Azevedo da Silva. (Org.). Understanding a Cancer Diagnosis.* https://novapublishers.com/shop/understanding-a-cancer-diagnosis/, 2020.

R.P. PINHEIRO, S.M.L. LIMA, D.M. SOUZA, and *et al.* Antivirus applied to jar malware detection based on runtime behaviors. *Scientific Reports - Nature: 12, 1945 (2022)*, 2022. doi: https://doi.org/10.1038/s41598-022-05921-5.

SANS. *SANS Institute InfoSec Reading Room. Out with The Old, In with The New: Replacing Traditional Antivirus.* Accessed on Feb 2020, 2017. URL `https://www.sans.org/reading-room/whitepapers/analyst/old-new-replacing-traditional-antivirus-37377`.

J. SU and *et al.* VASCONCELLOS, D. Lightweight classification of iot malware based on image recognition. *2018 IEEE 42nd Annual Computer Software and Applications Conference (COMPSAC)*, 2018. doi: https://doi.org/10.1109/COMPSAC.2018.10315.

M. WOZNIAK and J. *et al.* SILKA. Recurrent neural network model for iot and networking malware threads detection. *IEEE Transactions on Industrial Informatics.*, 2015. doi: https://doi.org/10.1109/TII.2020.3021689.

C. XIANG, S. Q. DING, and T. H. LEE. Geometrical interpretation and architecture selection of mlp. *The IEEE Transactions on Neural Networks and Learning Systems*, 16:84–96, 2005. doi: https://doi.org/10.1109/TNN.2004.836197.