UNIVERSIDADE ESTADUAL PAULISTA - UNESP

Instituto de Biociências, Letras e Ciências Exatas- Campus de São José do Rio Preto

JOÃO RAFAEL GREGÓRIO

Detecção de Domínios Gerados por Algoritmos com Aprendizado Profundo Incremental e DNS Passivo

São José do Rio Preto

2025

João Rafael Gregório

Detecção de Domínios Gerados por Algoritmos com Aprendizado Profundo Incremental e DNS Passivo

Dissertação, apresentada à Universidade Estadual Paulista (UNESP), Instituto de Biociências, Letras e Ciências Exatas, São José do Rio Preto, para obtenção do título de Mestre em Ciência da Computação.

Área de Concentração: Computação Aplicada

Orientador: Profº Dr. Adriano Mauro Cansian

Coorientador:

São José do Rio Preto

2025

IMPACTO POTENCIAL DESTA PESQUISA

O impacto potencial desta pesquisa reside na melhoria da detecção de domínios maliciosos gerados por algoritmos em ambientes reais, fortalecendo a defesa contra *botnets* e infraestruturas de ataque baseadas em DNS. A combinação de técnicas de aprendizado profundo com atualização incremental garante adaptação contínua a novas ameaças e preservando a eficácia do modelo ao longo do tempo. Assim, a proposta contribui diretamente para a segurança cibernética, oferecendo uma solução prática e escalável para proteção de redes corporativas e infraestruturas críticas.

POTENTIAL IMPACT OF THIS RESEARCH

The potential impact of this research lies in improving the detection of malicious domains generated by algorithms in real-world environments, strengthening defense against botnets and DNS-based attack infrastructures. The combination of deep learning techniques with incremental updating ensures continuous adaptation to new threats while preserving the model's effectiveness over time. Thus, the proposal contributes directly to cybersecurity, offering a practical and scalable solution for protecting corporate networks and critical infrastructures.

JOÃO RAFAEL GREGÓRIO

DETECÇÃO DE DOMÍNIOS GERADOS POR ALGORITMOS COM APRENDIZADO PROFUNDO INCREMENTAL E DNS PASSIVO

Dissertação apresentado(a) à Universidade Estadual Paulista (UNESP), Instituto de Biociências, Letras e Ciências Exatas, São José do Rio Preto, para obtenção do título de Mestre em Ciência da Computação.

Área de Concentração: Computação Aplicada

Data de defesa: 20/08/2025

BANCA EXAMINADORA

_____

Profº Dr. Adriano Mauro Cansian
UNESP – Instituto de Biociências, Letras e Ciências Exatas – Campus de São José do Rio Preto

_____

Profº Dr. Robson de Oliveira Albuquerque
Universidade de Brasília - UnB

_____

Profº Dr. Lucas Correia Ribas
UNESP – Instituto de Biociências, Letras e Ciências Exatas – Campus de São José do Rio Preto

Este trabalho é dedicado às crianças adultas que,
quando pequenas, sonharam em se tornar cientistas.

AGRADECIMENTOS

Agradeço imensamente à minha esposa Fabi pela compreensão e apoio incondicionais.

Ao meu orientador, Prof. Dr. Adriano Mauro Cansian, pela confiança, pelas direções apontadas e pelo caminho trilhado até aqui.

Aos professores das disciplinas de mestrado pelas orientações e conteúdos sempre muito úteis ao desenvolvimento deste trabalho.

Aos colegas das disciplinas de mestrado com os quais pude trocar experiências e conhecimento.

O presente trabalho foi realizado no âmbito do Projeto de Pesquisa e Inovação Tecnológica "Aprimoramento na Qualidade de Registro de Domínios Via Detecção Precoce de Conduta Abusiva - Fase 2", realizado com o apoio do Núcleo de Informação e Coordenação do Ponto BR (NIC.br) - Processo FUNDUNESP No. 3467/2023 (1º. Termo Aditivo 2025 – 2027). Agradecemos ao NIC.BR, à FUNDUNESP – Fundação de Amparo à Pesquisa da UNESP e à AUIn – Agência UNESP de Inovação, pelo suporte.

E a todos que de alguma maneira contribuíram para o desenvolvimento deste trabalho.

Muito obrigado!

“Se você conhece o inimigo e conhece a si mesmo, não precisa temer o resultado de cem batalhas.“

(Tzu, 2002)

RESUMO

O *Domain Name System* (DNS) é um dos serviços mais importantes da internet, responsável por mapear nomes de domínio em endereços IP. Embora seja amplamente utilizado de forma legítima, o DNS também pode ser explorado por cibercriminosos como parte de suas infraestruturas de ataque. Entre as mais comuns estão as *botnets*, redes de dispositivos comprometidos por *malware* e controlados remotamente de maneira coordenada. O controle dessas redes ocorre por meio de servidores de comando e controle (C2), que permitem aos atacantes gerenciar os dispositivos infectados. Para ocultar o endereço real dos C2 ou possibilitar a alteração frequente desses endereços sem interromper a comunicação com a botnet, os atacantes recorrem a algoritmos geradores de domínios (DGA), responsáveis por criar nomes de domínio pseudoaleatórios utilizados na comunicação maliciosa. Este trabalho apresenta um modelo de aprendizado profundo capaz de detectar domínios DGA utilizando técnicas de processamento de linguagem natural (NLP) para a classificação de textos curtos, além de uma metodologia de atualização incremental que permite incorporar novos exemplos, preservando a capacidade do modelo de identificar famílias emergentes de ameaças. A validação do modelo foi realizada em ambiente real, por meio da coleta de consultas DNS em rede local com DNS passivo, e complementada com o desenvolvimento de um painel de monitoramento DNS, destinado a acompanhar os domínios classificados como suspeitos pelo modelo. Nos experimentos, o modelo alcançou métricas expressivas tanto em ambiente controlado quanto em cenários reais, obtendo acurácia de 98,00%, precisão de 97,96%, recall de 97,95% e taxa de falsos positivos de 2,39%. O treinamento incremental demonstrou eficácia em evitar o esquecimento catastrófico, mantendo o desempenho estável ao longo do tempo. A validação em tráfego DNS de mundo real reforça a relevância do modelo na detecção de domínios DGA proposto neste trabalho, contribuindo significativamente para a segurança cibernética.

Palavras-Chave: DGA; botnet; DNS passivo; aprendizado profundo; redes neurais convolucionais.

ABSTRACT

The Domain Name System (DNS) is one of the most important services of the Internet, responsible for mapping domain names to IP addresses. Although it is widely used for legitimate purposes, DNS can also be exploited by cybercriminals as part of their attack infrastructures. Among the most common are botnets, networks of devices compromised by malware and remotely controlled in a coordinated manner. The control of these networks is carried out through command and control (C2) servers, which allow attackers to manage the infected devices. To conceal the real address of C2 servers or to enable frequent changes of these addresses without disrupting communication with the botnet, attackers employ Domain Generation Algorithms (DGA), which generate pseudo-random domain names used for malicious communication. This work presents a deep learning model capable of detecting DGA domains using natural language processing (NLP) techniques for the classification of short texts, in addition to an incremental update methodology that allows incorporating new examples, preserving the model's ability to identify emerging families of threats. The model was validated in a real-world environment through the collection of DNS queries on a local network using passive DNS monitoring, and complemented with the development of a DNS monitoring dashboard to track domains classified as suspicious by the model. In the experiments, the model achieved significant results in both controlled and real-world scenarios, reaching an accuracy of 98.00%, precision of 97.96%, recall of 97.95%, and a false positive rate of 2.39%. Incremental training proved effective in preventing catastrophic forgetting, maintaining stable performance over time. Validation with real-world DNS traffic reinforces the relevance of the proposed model in detecting DGA domains, making a significant contribution to cybersecurity.

Keywords: DGA; botnet; passive DNS; deep learning; convolutional neural networks.

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

A/V	Anti Vírus
C2	Comando e Controle
CNN	Convolutional Neural Network
CVE	Common Vulnerabilities and Exposures
DGA	Domain Genetarion Algorithm
DNN	Dense Neural Network
DNS	Domain Name System
FQDN	Fully Qualified Domain Name
HTTP	Hypertext Transfer Protocol
ICANN	Internet Corporation for Assigned Names and Numbers
IP	Internet Protocol
IRC	Internet Relay Chat
LSTM	Long Short-term Memory
OCR	Optical Character Recognition
P2P	Peer to Peer
PLN	Processamento de Linguagem Natural
RNN	Recurrent Neural Network
RFC	Request for Comments
TLD	Top Level Domain
UDP	User Datagram Protocol

SUMÁRIO

1 INTRODUÇÃO

Segundo dados da Datareportal Digital 2025: Global Overview Report (Kepios Pte. Ltd., 2025), cerca de 68% da população mundial está conectada à internet de alguma forma, significa dizer que mais de 5.5 bilhões de pessoas usam internet para os mais variados fins regularmente. Se olharmos para o uso empresarial, cerca de 93% das empresas estão conectadas à rede mundial, no Brasil 75% afirmam possuir algum tipo de presença online, seja rede social, website ou blog por exemplo.

Dados estatísticos do IX.br (Núcleo de Informação e Coordenação do Ponto BR, 2025a) demonstram que no Brasil, considerando todos os estados, a média diária de tráfego de internet passa dos 18Tbps. Segundo o portal Statista.com (Statista, 2023), no quarto trimestre de 2022 foram cerca de 349 milhões de nomes de domínio registrados em todos os domínios de primeiro nível, *Top Level Domain* (TLD). Somente para o TLD .COM foram mais de 200 milhões. No Brasil, segundo dados do Registro.br (Núcleo de Informação e Coordenação do Ponto BR, 2025b), já existem mais de 5,5 milhões de domínios .BR registrados, sendo aproximadamente 135 mil registros entre julho e agosto de 2025.

Diante deste cenário de números gigantescos podemos vislumbrar a quantidade de dados valiosos trafegados diariamente na internet mundial, de fotos a transações bancárias, de documentos de texto a sistemas e bancos de dados contendo dados pessoais de milhares de pessoas. É neste oceano de informação e tecnologia que grupos de hackers e ciber-criminosos buscam a todo tempo roubar informações, aplicar golpes, sequestrar dados e obter ganhos financeiros e/ou estratégicos. Uma das principais ferramentas utilizadas por esses grupos são as botnets (Schiller et al., 2011), aglomerados de computadores e dispositivos infectados por malwares que possibilitam seu controle remoto, seja para a orquestração de um ataque de negação de serviço distribuído, *Distributed Denial of Service* (DDoS) ou para o vazamento sistemático de informações de redes corporativas e/ou governamentais (Kambourakis et al., 2019).

Esses aglomerados de computadores e dispositivos infectados são controlados por meio de servidores de comando e controle (C2) conectados à internet (Kambourakis et al., 2019; Schiller et al., 2011). A fim de ofuscar a verdadeira localização dos servidores C2 e permitir sua mudança de endereço IP sem que haja perda de comunicação com os membros da botnet, a maioria dos malwares são desenvolvidos com algoritmos geradores de domínio, *Domain Generation Algorithm* (DGA).

Os DGAs presentes nos malwares geram uma quantidade significativa de nomes de domínio usados para a comunicação entre o cliente infectado e o servidor de comando e controle (C2) do atacante (Ravi et al., 2023). Devido ao enorme volume de nomes de domínio gerados por essas ameaças, a manutenção de listas com nomes de domínio identificados passa a ser ineficiente por dois motivos; primeiramente porque para que um nome de domínio seja incluído em uma lista ele precisa ser denunciado por alguém que tenha percebido sua atividade maliciosa, o que

demanda certo tempo; segundo porque a quantidade de nomes gerados por esses malwares é absolutamente grande, na ordem de centenas de milhares, o que certamente geraria um problema de desempenho para soluções de firewall por exemplo.

Nesse sentido, este trabalho se debruça sobre o problema de detecção automática dos DGA com técnicas de aprendizado profundo, a fim de identificar sua ocorrência em tráfego de rede de mundo real, além de verificar a viabilidade da aplicação de técnicas de aprendizado incremental a fim de manter o modelo detector atualizado com novos exemplos e novas famílias quando estas surgirem.

1.1 MOTIVAÇÃO

Em redes corporativas, a presença de consultas de resolução de nomes de domínios gerados por algoritmo pode significar a presença de computadores infectados por malwares que podem vazar dados sigilosos, promover um ataque de negação de serviço ou mesmo criptografar arquivos e bancos de dados completos a fim de exigir resgates financeiros (*ransomware*[1]), portanto identificar esse tipo de consulta de maneira precoce significa antecipar-se a incidentes graves de cibersegurança.

Em um breve levantamento podemos elencar algumas ameaças importantes, com impacto global, responsáveis por enormes prejuízos financeiros que utilizaram em sua infraestrutura de ataque algum tipo de DGA.

Um dos ataques recentes mais relevantes e que utilizou DGA em algum momento de seu ciclo de vida foi o ataque à SolarWinds (Kruti; Butt; Sulaiman, 2023) ocorrido em 2020, este ganhou notoriedade quando hackers conseguiram obter acesso não-autorizado ao código fonte do sistema Orion, usado para monitoramento de infraestrutura tecnológica. Mais de 18 mil clientes da SolarWinds instalaram as versões infectadas pelo malware Sunburst, incluindo gigantes da tecnologia como Microsoft, Intel e Cisco. O malware Sunburst utilizou-se de um mecanismo elaborado de DGA (Wong, 2023) que, além de manter a comunicação da estação infectada com os servidores C2 ainda era capaz de identificar com precisão de qual estação infectada as informações obtidas eram provenientes.

Outro caso relevante é o da botnet Mirai, originalmente criada para ataques DDoS contra servidores de Minecraft (Antonakakis et al., 2022), que após seu código fonte ser publicado em 2017, passou a se espalhar atacando principalmente dispositivos IoT como webcams, câmeras de segurança e roteadores (Azhari et al., 2022). Com histórico de ataques DDoS utilizando centenas de milhares de endereços IP, a botnet Mirai permanece ativa com novas variantes que se utilizam de vulnerabilidades conhecidas, como é o caso da variante IoT.Linux.MIRAI.VWISI encontrada em 2020, que se vale da CVE-2020-10173 para explorar roteadores domésticos (SegInfo, 2021). Esta botnet utiliza DGA em sua infraestrutura de manutenção e ofuscação de comunicação entre

[1] Software mal-intencionado que visa criptografar dados do dispositivo infectado a fim de que seja solicitado um valor de resgate para que a chave de decriptografia seja fornecida pelo atacante.

botclient e C2. Exemplos de domínios relacionados a esta ameaça são percebidos e reportados diariamente para órgãos de manutenção de listas de domínios maliciosos conhecidos.

O *ransonware* Cryptolocker originado em 2013, utilizava da botnet Gameover Zeus para se espalhar, este criptografou dados de milhares de usuários e empresas (Kamil et al., 2022) causando prejuízos da ordem de 27 milhões de dólares. Uma operação colaborativa internacional denominada Operação Tovar, que envolveu o FBI, o Depto. de Justiça dos Estados Unidos e a Europol, além de empresas da área de segurança cibernética como Dell SecureWorks, Microsoft e Abuse.ch foi capaz de mitigar a proliferação do Cryptolocker atuando sobre a botnet Gameover Zeus, derrubando temporariamente a comunicação com os servidores C2. Embora o Cryptolocker original tenha sido mitigado juntamente com a botnet Gameover Zeus, novos ransomwares surgiram, utilizando-se de infraestruturas baseadas em botnets para sua proliferação e controle. Segundo o portal Ciso Advisor (Ciso Advisor, 2023), o Brasil ocupava no final de 2022 o segundo lugar no ranking mundial de *ransonware*, sendo alvo de cerca de 30% de todos os ataques *ransonware* em todo o mundo.

Percebe-se portanto que ameaças cibernéticas importantes se utilizam de DGA em sua infraestrutura de ataque e manutenção de comunicação. Nesse sentido, a identificação automatizada de DGAs, por meio de modelos de inteligência artificial e aprendizado profundo representa uma necessidade em razão do volume de domínios registrados diariamente, além de permitir a identificação de um DGA em sua primeira consulta, muito antes que este seja adicionado às listas de bloqueio ou servidores DNS Sinkhole [2] por exemplo.

1.2 JUSTIFICATIVA

Na literatura podemos encontrar diversos trabalhos focados na identificação de domínios gerados por algoritmos utilizando aprendizado profundo (Park et al., 2022; Sun; Liu, 2023; Tuan; Long; Taniar, 2022; Huang et al., 2022), estes trabalhos podem ser organizados quanto às fontes dos dados e quanto à técnica de extração de características utilizada.

Alguns trabalhos se utilizam de listas de domínios já rotulados vindos de fontes como Majestic Million (Majestic, 2023) para domínios legítimos e Bambenek Consulting (Bambenek Consulting OSINT, 2023), DGArchive (Fraunhofer FKIE, 2023) e NetLab360 (NetLab 360, 2022) para exemplos de domínios gerados por algoritmo (Gogoi; Ahmed, 2023; Huang et al., 2022). Um segundo tipo de abordagem quanto à fonte dos dados são os trabalhos que lançam mão de dados coletados em servidores DNS Passivo (Silveira et al., 2021) em um espaço determinado de tempo (Bao; Wang; Lan, 2020), utilizando-se de listas apenas para a rotulagem de dados de treinamento iniciais.

Quanto à técnica de extração de características, alguns trabalhos se utilizam de características léxicas dos nomes de domínio e outras características extraídas de dados Whois e DNS passivo (Sun; Liu, 2023). Outros trabalhos buscam extrair características com técnicas de aprendi-

[2] Servidores DNS configurados para não rotear determinados nomes de domínio considerados maliciosos.

zado profundo (Tuan; Long; Taniar, 2022), como redes neurais convolucionais, embedding e autoencoders.

Nota-se que todos esses trabalhos têm em comum o uso de conjuntos de dados fixos para treinamento, validação e teste de seus modelos de aprendizado de máquina. Porém, como novas ameaças surgem diariamente e os malwares evoluem constantemente a fim de evadirem camadas de segurança, faz-se necessário um modelo capaz de se atualizar a partir de novos dados, sem que haja a necessidade de se treinar novamente o modelo com todo o volume de dados coletados até aquele momento, dados que muitas vezes nem estão mais disponíveis.

Sendo assim, um modelo de aprendizado profundo incremental tende a se manter com níveis de desempenho ideais por mais tempo e irá gradativamente passar a identificar novas ameaças conforme estas forem identificadas e retro-alimentadas ao modelo. A integração de um modelo profundo de aprendizado incremental a um servidor de DNS Passivo permite, por outro lado, a identificação de domínios gerados por algoritmo em tráfego de mundo real, inclusive com a identificação de exemplos que ainda não foram adicionados a listas de bloqueio, o que caracteriza uma vantagem estratégica para equipes de segurança cibernética.

1.3 OBJETIVOS

Este trabalho tem por objetivo apresentar um modelo de aprendizado profundo incremental baseado em redes neurais convolucionais e processamento de linguagem natural para identificar e classificar domínios gerados por algoritmo de maneira precoce. O modelo desenvolvido deve ser capaz de identificar um domínio gerado por algoritmo utilizando-se exclusivamente do nome de domínio de segundo nível, sem que haja o desenvolvimento de características adicionais ou quaisquer outros dados complementares.

De maneira geral, os objetivos específicos alcançados como resultados deste trabalho são:

- Desenvolver um modelo de detecção de DGAs utilizando aprendizado profundo e processamento de linguagem natural.
- Aplicar técnicas de aprendizado incremental ao modelo desenvolvido.
- Integrar o modelo desenvolvido com uma solução de DNS Passivo para monitoramento em ambiente de mundo real.
- Comparar resultados obtidos pelo modelo proposto em laboratório e em dados mundo real.
- Construir um painel de monitoramento de consultas DNS utilizando o modelo desenvolvido e a ferramenta Grafana (Grafana Labs, 2023).

1.4 CONTRIBUIÇÕES

As contribuições alcançadas durante as pesquisas realizadas para a produção deste trabalho foram:

- Desenvolvimento de um modelo de aprendizado profundo utilizando redes neurais convolucionais e processamento de linguagem natural para detecção de DGAs.
- Aplicação de técnicas de aprendizado incremental ao modelo proposto, verificando a viabilidade, as vantagens e limitações dessa técnica de treinamento de redes neurais artificiais.
- Aplicação do modelo desenvolvido em ambiente de mundo real, com a integração deste a uma solução de DNS Passivo.
- Comparação dos resultados obtidos pelo modelo proposto em laboratório e em dados de mundo real.
- Produção de um painel de monitoramento de consultas DNS no âmbito da UNESP utilizando o modelo desenvolvido.
- Um artigo intitulado “Deep Convolutional Neural Networks and Character-Level Embeddings for DGA Detection” apresentado na 26th ICEIS 2024, conferência internacional Qualis A3.
- Um artigo intitulado "Class Incremental Deep Learning: A Computational Scheme to Avoid Catastrophic Forgetting in Domain Generation Algorithm Multiclass Classification"publicado na revista MDPI Applied Sciences, revista com fator de impacto 2.5, Qualis A2.

1.5 ORGANIZAÇÃO DO TRABALHO

Este trabalho está organizado da seguinte forma: o Capítulo 1, que inclui esta Seção, apresenta as motivações, justificativas e objetivos do trabalho.

O Capítulo 2 apresenta a fundamentação teórica que suporta este trabalho de pesquisa, os conceitos de aprendizado profundo mais importantes para este trabalho, além dos trabalhos recentes relacionados ao tema desta pesquisa, a fim de apresentar o estado da arte da detecção e classificação de domínios gerados por algoritmos com técnicas de aprendizado profundo.

No Capítulo 3, as metodologias, fontes de dados, o pré-processamento dos dados e os modelos de aprendizado profundo propostos nesta pesquisa, bem como a metodologia de avaliação dos modelos em dados de mundo real com DNS Passivo. Ainda no Capítulo 3 algumas conclusões parciais obtidas durante os primeiros passos desta pesquisa.

O Capítulo 4 apresenta os resultados obtidos nos testes realizados em cada etapa da pesquisa, além de uma comparação direta dos resultados dos modelos propostos com os resultados de modelos apresentados pela literatura recente. Ainda no Capítulo 4 são apresentados os resultados obtidos sobre os dados coletados em ambiente de mundo real por meio de DNS Passivo.

Finalmente, no Capítulo 5 apresentamos as discussões, avanços e limitações, conclusões e sugestões para trabalhos futuros.

2 FUNDAMENTAÇÃO TEÓRICA

Este capítulo apresenta os fundamentos teóricos utilizados neste projeto, as definições técnicas a respeito do serviço DNS, um *overview* sobre botnets e domínios gerados por algoritmos, o estado da arte das pesquisas que se debruçam sobre essa questão e as tecnologias mais atuais de aprendizado profundo utilizadas neste projeto.

2.1 SISTEMA DE NOMES DE DOMÍNIO (DNS)

Da mesma forma que é muito mais fácil para o ser humano se lembrar do nome de uma pessoa em vez de se lembrar do seu número de Cadastro de Pessoa Física (CPF), ou se lembrar do nome de uma determinada empresa em vez de seu Cadastro Nacional de Pessoa Jurídica (CNPJ), também é muito mais simples decorar o nome de um determinado website em vez do endereço IP do servidor onde este se encontra.

Apesar de ser perfeitamente possível se acessar um website ou qualquer outro serviço na internet utilizando-se do endereço IP, essa prática não é o melhor caminho por algumas razões, e a primeira delas, exposta acima, nem é a mais importante. Além de ser muito mais simples se lembrar do nome de um website em vez de seu endereço de IP, também devemos levar em conta que uma empresa pode precisar mudar o servidor onde seu website está hospedado, mudando também seu endereço IP, portanto a partir desse momento, todos que se utilizam do website deveriam ser informados da mudança, sob o risco de não conseguirem mais acesso.

Atualmente, empresas que possuem websites com demandas muito grandes de acesso costumam distribuir a carga de acesso mantendo servidores distribuídos geograficamente com seu conteúdo sincronizado. O serviço DNS entra em ação resolvendo um determinado endereço IP para acessos vindos do Brasil, enquanto entrega outro endereço IP para acessos vindos dos Estados Unidos, por exemplo. Dessa forma o acesso é garantido para ambos os públicos sem que haja sobrecarga de apenas um servidor.

Na ARPANET já se havia percebido que havia a necessidade de tornar o acesso aos conteúdos mais amigável ao ser humano, portanto naquele ambiente havia um arquivo hosts.txt com os endereços IP e nomes relacionados a eles, diariamente os demais servidores copiavam esse arquivo para se manterem atualizados quanto aos nomes e endereços IP da rede completa (Tanenbaum; Feamster; Wetherall, 2021).

Essa estratégia funcionaria bem enquanto o número de hosts da rede se restringisse a algumas centenas, porém seria colocada contra as cordas e ficaria impraticável com o aumento de hosts para alguns milhares ou os milhões que existem atualmente. Nesse sentido, em 1983 foi criado o primeiro Sistema de Nomes de Domínios (DNS), sendo este um dos serviços mais importantes da rede mundial de computadores, mapeando nomes de domínio em endereços IP. O DNS tem evoluído desde então, recebendo diversas atualizações como pelas RFCs 1034, 1035, 2181, tanto

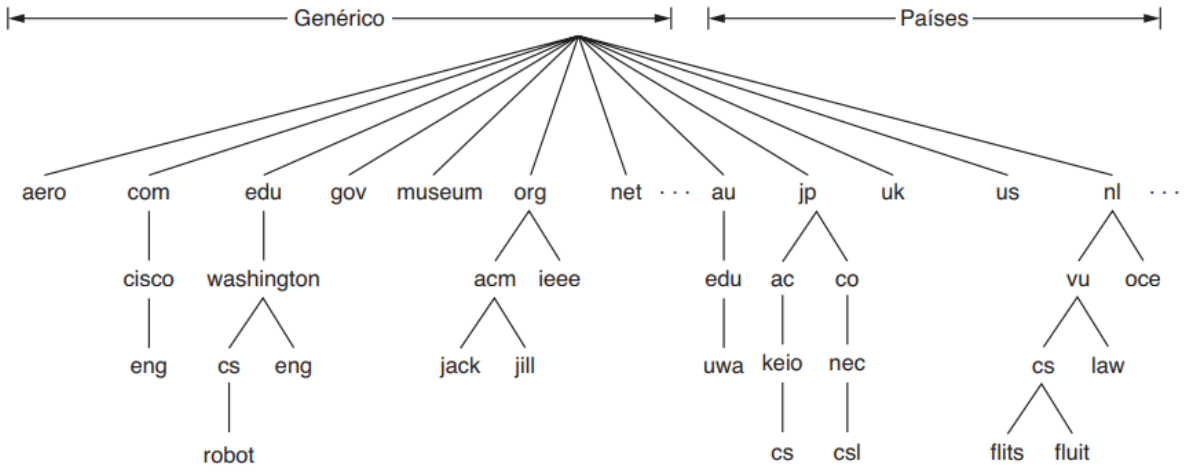
na ampliação de domínios quanto na implantação de sistemas de segurança como o DNSSec definido pelas RFCs 4033, 4034 e 4035, por exemplo.

2.1.1 Sistema hierárquico de nomes de domínio

Semelhantemente ao sistema postal, que atribui nomes ou códigos para países, estados, cidades e ruas, a fim de identificar um único endereço em uma determinada localização geográfica, o sistema de nomes de domínios aplica uma estrutura hierárquica para definir os nomes de domínio visando não haver conflitos entre nomes (Tanenbaum; Feamster; Wetherall, 2021).

A ICANN (*Internet Corporation for Assigned Names and Numbers*), organização criada em 1998 é a responsável por manter a hierarquia dos TLDs e controlar a criação de novos TLDs. Entre os TLDs os mais comuns estão os gTLD (*generic Top Level Domain*) e os ccTLD (*country code Top Level Domain*), a Figura Figura 1 apresenta parte dos domínios genéricos originais e outros introduzidos posteriormente pela ICANN. Atualmente cada país possui seu próprio TLD conforme definido na ISO 3166 de 1997 e suas atualizações posteriores.

Figura 1 – Parte do espaço de nomes de domínios.

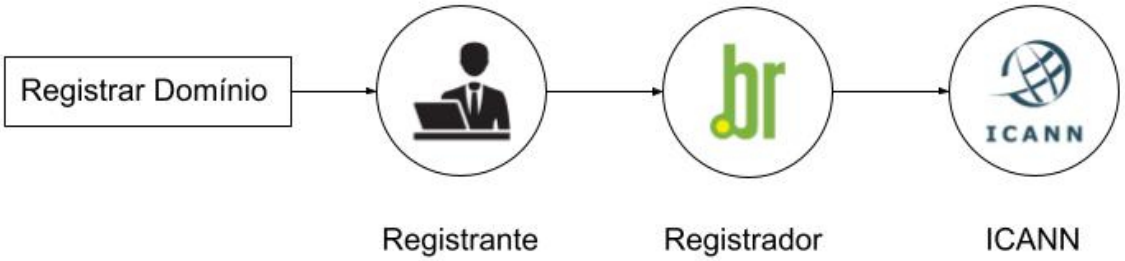


Fonte: (Tanenbaum; Feamster; Wetherall, 2021).

Enquanto a ICANN mantém o controle sobre as TLDs, sua distribuição ocorre por registradores apontados pela mesma, no Brasil o .BR e outros TLDs sob o .BR, são controlados pelo Registro.br (Núcleo de Informação e Coordenação do Ponto BR, 2025b). Nomes de segundo nível como minha-empresa.com.br, podem ser adquiridos com registrantes, empresas que oferecem este serviço cobrando um valor relativamente acessível pela realização do mesmo.

Sendo assim, os domínios de segundo nível podem ser registrados diretamente com diversas empresas como GoDaddy, NameCheap e Locaweb, sendo este um processo simples, rápido e relativamente barato. A Figura Figura 2 apresenta a relação entre ICANN, registrador e registrante de domínios para a internet.

Figura 2 – Processo de registro e relação entre ICANN, registrador e registrante.



Fonte: Adaptado de (Tanenbaum; Feamster; Wetherall, 2021).

Nomes abaixo do segundo nível como, por exemplo, www.minha-empresa.com.br ficam a cargo do próprio detentor do domínio de segundo nível, e normalmente representa uma única máquina ou um conjunto de máquinas com um mesmo propósito, o WWW é normalmente utilizado para apontar a máquina que hospeda o site web da empresa. Estes nomes completos são conhecidos como Nomes de Domínio Totalmente Qualificados (FQDN).

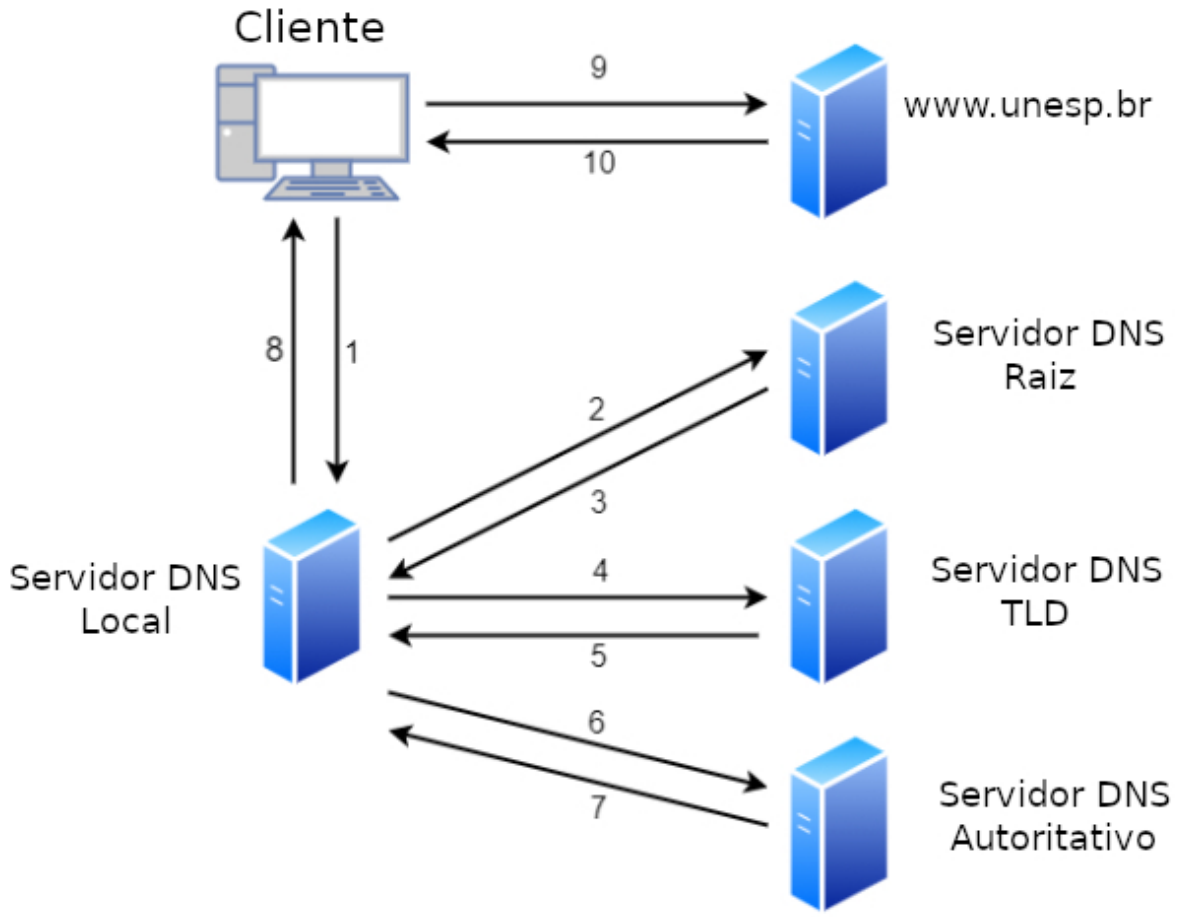
Diariamente, milhares de domínios de segundo nível são registrados dessa forma, para os mais variados fins, seja para hospedar o site de uma nova empresa ou outro fim legítimo, seja para hospedar páginas falsas ou utilizar o nome registrado em um servidor C2 de uma botnet. Esse assunto será tratado em maiores detalhes na Seção 2.2.

2.1.2 Mecanismo de consulta DNS

O processo de resolução de nomes de domínio é um processo hierárquico, que conta com uma sequência de consultas feitas em servidores DNS a fim de retornar o endereço IP mapeado para um nome de domínio. Quando uma aplicação precisa mapear um nome de domínio em um endereço IP esta dispara uma consulta para o servidor DNS local, informando o nome de domínio que deseja resolver.

O servidor DNS local, também chamado de resolvedor recursivo local, dispara uma consulta para um conjunto de servidores DNS responsáveis por cada uma das partes da hierarquia de nomes, cada servidor de nomes responsável por uma parte da hierarquia é chamado de servidor de nomes autorizado ou autoritativo (Tanenbaum; Feamster; Wetherall, 2021). A sequência de consultas para resolução de um nome de domínio totalmente qualificado é demonstrada pela Figura Figura 3.

Figura 3 – Consulta por um nome de domínio totalmente qualificado.



Fonte: Adaptado de (Sartorello, 2023).

Apesar do tipo de consulta DNS mais comum ser as do tipo A, que buscam mapear um nome de domínio para um endereço IPv4, há também outros tipos de consulta DNS, como o tipo AAAA que espera como resposta um endereço de IPv6 ou Nome Canônico, *Canonical Name* (CNAME), que mapeia um nome de domínio para outro nome de domínio. A Tabela 1 apresenta os tipos de consultas DNS, seus significados e os valores esperados como resposta.

Tabela 1 – Principais tipos de registros DNS.

Tipo	Significado	Valor
SOA	Início de autoridade	Parâmetros para essa zona
A	Endereço IPv4	Inteiro de 32 bits
AAAA	Endereço IPv6	Inteiro de 128 bits
MX	Troca de mensagens de e-mail	Prioridade, domínio aceitando correio eletrônico
NS	Servidor de nomes	Nome de um servidor para este domínio
CNAME	Nome canônico	Nome de domínio
PTR	Ponteiro	Nome alternativo de um endereço IP
SPF	Política do transmissor	Codificação da política de envio de e-mail
SRV	Serviço	Host que oferece o serviço
TXT	Texto	Texto ASCII descritivo

Fonte: (Tanenbaum; Feamster; Wetherall, 2021).

Como podemos observar, o serviço DNS atualmente realiza muito mais do que a tradução de um nome em IP, como havia sido projetado no início de seu desenvolvimento. Dessa forma, nota-se que o DNS é um dos serviços fundamentais para a rede mundial de computadores cuja evolução acompanhou o surgimento de novas necessidades, inclusive de segurança. Porém, apesar de toda a evolução, este serviço continua a ser usado de forma espúria para diversas finalidades ilegais.

2.1.3 DNS Passivo

Um servidor DNS fornece o serviço de resolução de nomes através de consultas recebidas e respondidas pela porta 53, normalmente UDP. Os pacotes referentes a essas consultas DNS realizadas pelos clientes podem ser capturadas por *sniffers* [1]. Um *sniffer* especializado em capturar e armazenar consultas DNS é chamado DNS passivo (Silveira et al., 2021).

Por se tratar de um conjunto de dados muito relevante para a identificação e triagem de ameaças cibernéticas diversas, o DNS Passivo tem sido utilizado em alguns países como forma de obter dados para pesquisa em cibersegurança e para servir de ferramenta para a prevenção e resposta a incidentes. Uma iniciativa importante é mantida pelo governo de Luxemburgo, com o Computer Incident Response Center Luxembourg (CIRCL) Passive DNS 2.0.

Mantida pela Universidade de Indiana, o Research and Education Networks Information Sharing and Analysis Center (REN-ISAC) é outra iniciativa que aplica DNS Passivo, esta conta com mais de 700 membros, compostos por universidades quem mantém pesquisas na área de cibersegurança e resposta a incidentes.

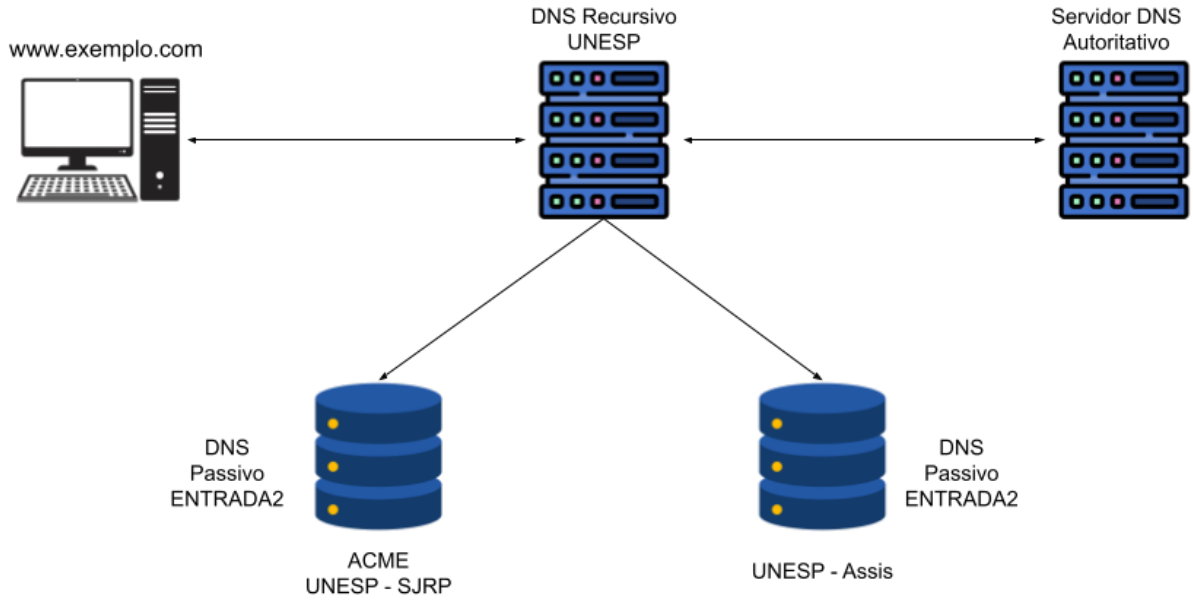
Outra iniciativa colaborativa é o DNS Operations, Analysis, and Research Center (DNS-OARC), que promove anualmente desde 2016 uma coleta de 48 horas de consultas DNS em servidores distribuídos pelo mundo todo, incluindo grandes empresas como GoDaddy, IBM; instituições como NASA e ICANN; além de universidades como a Universidade de Maryland, e em 2024 contou com a colaboração da UNESP em uma atividade desenvolvida junto ao Laboratório Acme de Pesquisa em Cibersegurança, onde este trabalho foi desenvolvido.

Neste trabalho utilizamos a solução de DNS Passivo ENTRADA 2 (Maarten Wullink, Moritz Muller, Marco Davids, Giovane C. M. Moura, and Cristian Hesselman, 2016), projeto desenvolvido e mantido pela *Stichting Internet Domeinregistratie Nederland* (SIDN), instituição holandesa responsável pelo registro de domínios sob o ccTLD .NL. Este sistema para coleta, enriquecimento e armazenamento de consultas DNS está atualmente instalado em um servidor no Câmpus Unesp Assis/FCL e outro no Câmpus de São José do Rio Preto/IBILCE nas instalações do Laboratório Acme de Pesquisa em Cibersegurança.

Estes recebem as consultas coletadas diretamente no servidor DNS recursivo da Unesp hospedado na Reitoria Unesp. A Figura Figura 4 apresenta o mecanismo de coleta de consultas DNS no âmbito da Unesp.

[1] Softwares especializados em capturar pacotes de dados em uma rede.

Figura 4 – Esquema de coleta de consultas DNS com DNS Passivo Entrada2 na Unesp.



Fonte: Elaborado pelo Autor.

Esta estrutura armazena atualmente uma janela de 12 meses de consultas DNS realizadas por estações de trabalho, computadores, telefones celulares e demais dispositivos conectados à rede da Unesp. Esses dados são utilizados em outros trabalhos de pesquisa sobre abuso de DNS desenvolvidos na universidade, por se tratar de uma base rica, de ambiente real, com dados de todas as etapas e tipos de consultas DNS, o que proporciona a abertura de um amplo leque de opções para abordagens de pesquisas em abusos de DNS.

2.2 BOTNETS

Uma das estruturas mais utilizadas por grupos hacker e ciber-criminosos são as botnets, uma coleção de dispositivos infectados por malware que podem ser controlados remotamente por meio de servidores C2. Segundo (Schiller et al., 2011) existem algumas características que definem uma botnet em função de seu comportamento e orquestração, a saber:

- Uma botnet deve possuir pelo menos um botserver e um botclient (normalmente são milhares de botclients em uma botnet).
- Um botclient deve possuir um mecanismo de interpretação de comandos que possibilite seu controle remoto.
- Uma botnet deve ser capaz de agir coordenadamente, usando todos ou parte de seus botclients.

Já (Kambourakis et al., 2019) define uma botnet como uma infraestrutura de ataque formada por dispositivos conectados em forma de rede usando uma variedade de protocolos como HTTP,

IRC e protocolos P2P. Tanto (Schiller et al., 2011) quanto (Kambourakis et al., 2019) definem ciclos de vida para as botnets baseados em momentos distintos em relação à sua função e eventos relacionados. Para (Kambourakis et al., 2019) o ciclo de vida de uma botnet pode ser dividido em três fases:

- Recrutamento: estágio onde os dispositivos são explorados a fim de se inserir o malware para controle remoto, este estágio pode contar com ataques a vulnerabilidades conhecidas de sistemas, engenharia social ou mesmo dispositivos removíveis (pendrives, cartões SD).
- Estágio de Comando e Controle: neste estágio o objetivo é garantir a manutenção da comunicação do botclient com o botserver mediante a execução recorrente de comandos para manter os botclients atualizados com os códigos maliciosos de controle da botnet.
- Estágio de atividade da botnet: as atividades de uma botnet pode variar desde garantir o vazamento sistemático de informações e ataques distribuídos de negação de serviço até o escaneamento de outros dispositivos vulneráveis próximos a fim de recrutá-los, ampliando o alcance da botnet.

Em (Schiller et al., 2011) é definido um ciclo de vida de botnet um tanto quanto mais detalhado em nove etapas distintas:

- Dispositivo explorado se torna um botclient.
- Botclient se anuncia para que seja reconhecido na botnet.
- Instalação do módulo Anti Anti-Vírus.
- Implantação de segurança contra Anti-vírus, detecções de usuários ou intervenção de outro hacker.
- Contato com C2 para execução de comandos.
- Atualização de payloads.
- Execução de comandos.
- Informação de resultados ao C2.
- Limpeza de evidências e abandono do botclient.

A Figura Figura 5 apresenta uma junção de ambas as abordagens de ciclo de vida de botnet, com momentos distintos delimitados e momentos muito próximos em função aglutinados em um único item.

Figura 5 – Ciclo de vida de uma botnet.

Fonte: Elaborado pelo autor.

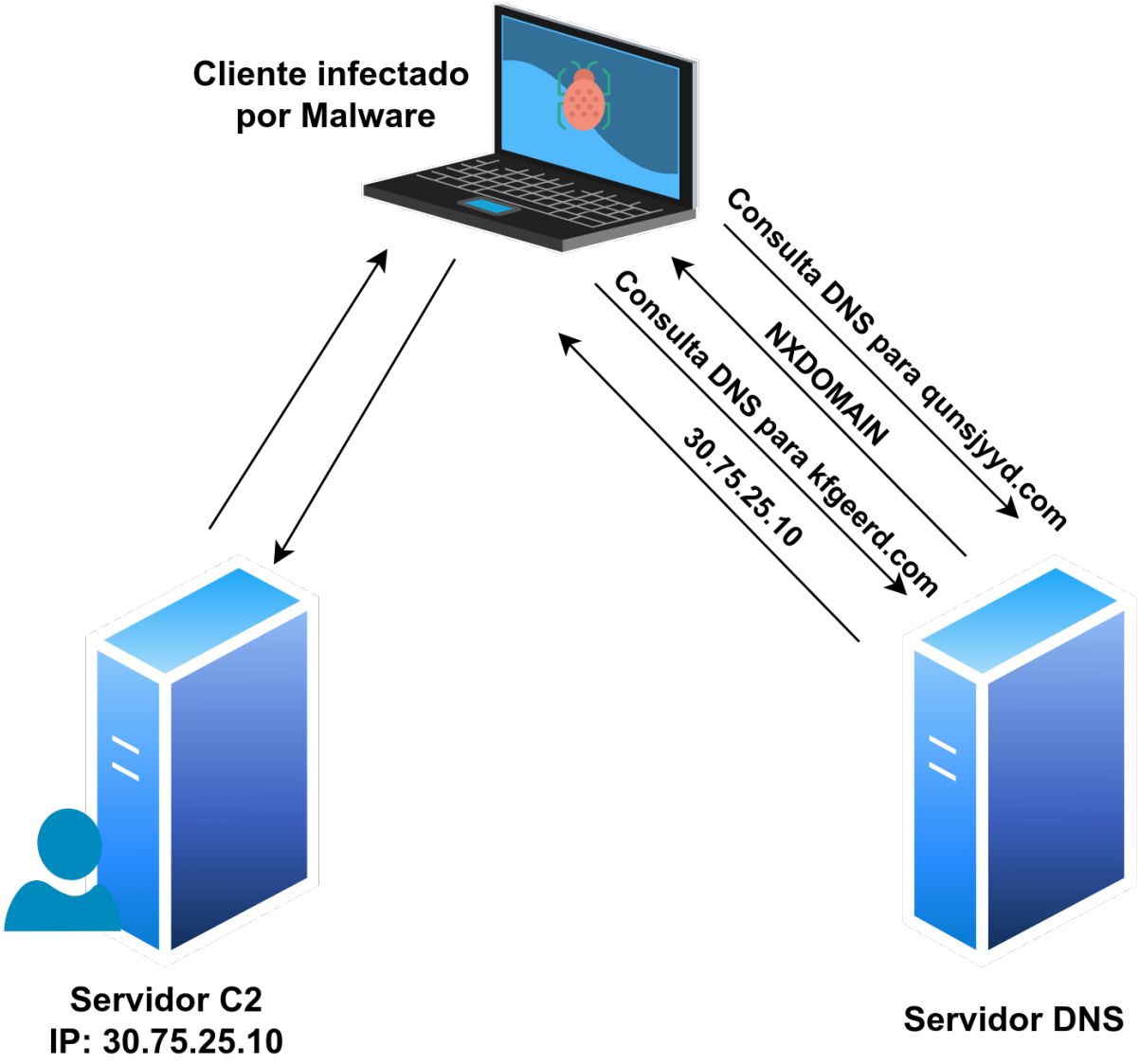
Comparando a Figura 5 com o modelo proposto por (Kambourakis et al., 2019), os itens 1, 2 e 3 compõem o estágio de Recrutamento; os itens 4 e 5 o estágio de Comando e Controle e itens 6 e 7 o estágio de Atividade. O item 8 traz do modelo proposto por (Schiller et al., 2011) o estágio de limpeza de evidências.

Percebe-se que, em todas as fases do ciclo de vida de uma botnet, a comunicação entre botclients e servidores C2 se faz necessária e fundamental; neste momento uma das estratégias para a manutenção da comunicação, evasão de mecanismos de segurança de rede e ofuscação das origens da comunicação são os DGA que veremos em detalhes na Seção 2.3 a seguir.

2.3 DOMÍNIOS GERADOS POR ALGORÍTMOS

Como vimos anteriormente, a manutenção da comunicação entre botclient e servidor C2 constitui parte fundamental da infraestrutura de ataque distribuída que é a própria botnet. A fim de flexibilizar a alteração de endereço IP dos servidores C2, permitindo assim que um IP denunciado por abuso possa rapidamente ser substituído por outro sem restrições, ou para evadir sistemas de segurança de rede e ofuscar a verdadeira localização dos servidores C2, a maioria dos malwares utilizados em botnets são construídos com algoritmos geradores de domínio, programas que geram de forma pseudo-randômica nomes de domínio que são registrados pelos detentores da botnet para serem apontados para seus servidores C2 (C et al., 2022). A Figura Figura 6 apresenta o esquema de uso de domínios gerados por algoritmos por botclient a fim de manter a comunicação com o servidor C2 da botnet.

Figura 6 – Esquema de uso de domínio gerado por algorítmo por botclient.



Fonte: Adaptado de (Shahzad; Sattar; Skandaraniyam, 2021).

Quando um botclient precisa se comunicar com seu servidor C2 o algoritmo gerador de domínios gera um domínio compatível com os registrados pelo detentor da botnet e então o dispositivo infectado solicita a resolução do nome de domínio gerado, caso este não retorne um IP o algoritmo gera outro nome de domínio compatível com seu escopo, até que haja o devido encaminhamento para o IP do servidor C2.

Cada malware possui seu próprio mecanismo de geração de nomes de domínio, construindo estes de acordo com suas especificidades, com sequências de caracteres e comprimentos distintos. Em (Fraunhofer FKIE, 2023) estão catalogadas mais de 120 famílias de malwares com seus respectivos exemplos de domínios gerados por algoritmos, cada um com seu comprimento, TLDs e morfologia. Basicamente as famílias de algoritmos geradores de domínio podem ser organizadas em três tipos quanto à morfologia dos nomes de domínio gerados; pseudo-randômicos, baseados em domínios semente e baseados em listas de palavras (Fraunhofer FKIE, 2023).

Os nomes de domínio pseudo-randômicos são apresentados como uma sequência de letras e/ou números ininteligíveis para o ser humano, esse tipo representa a imensa maioria dos exemplos domínios gerados por algoritmos encontrados nas bases de dados disponíveis.

Nomes de domínio gerados por algoritmo baseados em listas de palavras normalmente usam duas ou mais listas de palavras, geralmente em idioma inglês, e geram os nomes de domínio usando a permutação dessas palavras, esse tipo de domínio gerado por algoritmo é mais difícil de ser detectado, pois mimetiza a morfologia dos domínios legítimos (Yang et al., 2019).

Já os domínios gerados por algoritmo baseados em um domínio semente partem de um nome de domínio existente e aplicam alterações a este, como a adição de caracteres no início ou no final do nome (Fraunhofer FKIE, 2023), esse tipo também é mais complexo de ser identificado devido à sua semelhança morfológica com os domínios legítimos. A Tabela 2 apresenta algumas famílias de malware/botnet, seu tipo de domínio gerado por algoritmo e alguns exemplos dos mesmos.

Tabela 2 – Famílias DGA por tipo de formação do nome de domínio e exemplos.

Família	Tipo DGA	Exemplos
Suppobox	Lista de palavras	perhapsmeasure.net windowafraid.net wintermeasure.net
Matsnu	Lista de palavras	saladdoctortrainer.com strangerbiketechnology.com termacceptyear.com
Mirai	Pseudo-randômico	vmdefmnsdoj.tech xpknpxmywqsr.online oornsduuwjli.tech
Gameover	Pseudo-randômico	1pb98u4egqbcwzes185mpfyvc.com 1phu3tw1xne48hy0s8df17ktgb0.net gyjcf918ifxjyi07gt011pu5k8.biz
Banjori	Domínio semente	fnramen.com ekwallaabettingk.com wuccinalcentricem.com
Darkshell	Domínio semente	243axvya.com u035zy.com S33Wwy.com

Fonte: Elaborado pelo autor.

Podemos perceber que, mesmo se tratando de um único tipo de ameaça, quando olhamos mais de perto notamos que há uma ampla variedade de tipos distintos desses domínios gerados por algoritmo e, portanto, a construção de modelos para detecção automatizada destes se torna um desafio importante para a segurança cibernética.

2.4 APRENDIZADO DE MÁQUINA E APRENDIZADO PROFUNDO

A Inteligência Artificial (IA) é a área de conhecimento que visa o desenvolvimento de sistemas que possuam a habilidade de realizar tarefas que, de alguma forma, exijam a capacidade de inteligência humana. Nos primórdios da IA esta se especializou em resolver problemas que eram difíceis para seres humanos, mas que eram relativamente fáceis para computadores, como problemas matemáticos, estes problemas são os que podem ser descritos facilmente com uma lista de passos ou uma função matemática (Goodfellow; Bengio; Courville, 2016).

Mas o grande desafio se encontra em produzir sistemas que consigam realizar tarefas que são simples para os seres humanos, mas que são muito complexos de se descrever em uma lista fixa de passos ou uma função matemática, como reconhecer a voz de alguém ou reconhecer o rosto de uma pessoa em uma imagem com diversos rostos (Goodfellow; Bengio; Courville, 2016).

O Aprendizado de Máquina (ML) é uma sub-área da IA que se concentra em desenvolver algoritmos capazes de aprender com dados fornecidos e decidir em função do reconhecimento de padrões e relacionamentos entre esses dados (Mukhamediev et al., 2021; Singapore Computer

Society, 2020). Os métodos de ML podem envolver Máquinas de Vetores de Suporte (SVM), Árvores de Decisão, Regressão, Redes Neurais Artificiais entre outros.

As Redes Neurais Artificiais são um sub-campo do ML, normalmente descritas como um conjunto de elementos conectados, chamados neurônios artificiais, organizados em camadas (Mukhamediev et al., 2021; Singapore Computer Society, 2020). O Aprendizado Profundo, portanto, é um sub-campo do ML que utiliza redes neurais artificiais para processar os dados de entrada em várias camadas de algoritmos sem qua haja interação humana na elaboração de características (Mukhamediev et al., 2021).

Em ML, devemos fornecer dados de entrada que representem bem o objeto analisado, estando a capacidade do algoritmo de aprendizado de máquina de resolver o problema muito atrelada ao nível de representação dos dados fornecidos. Portanto, no ML existe uma etapa conhecida como engenharia de atributos, onde são construídos os atributos que possam representar bem o objeto analisado (Goodfellow; Bengio; Courville, 2016).

Exemplificando, se estivermos analisando dados de pacientes para que o algoritmo de ML dê o diagnóstico da doença que o paciente tem, devemos fornecer dados como sintomas específicos, dados de exames realizados e idade do paciente. Ocorre que existem problemas para os quais não é tão simples construir características, como, por exemplo, o reconhecimento de um automóvel em uma imagem. É muito difícil descrever uma roda em função dos pixels de uma imagem, por exemplo.

Em Aprendizado Profundo, as características são extraídas pela própria rede neural artificial utilizada, sem que haja o processo de engenharia de atributos. Essas características extraídas pela rede neural profunda são denominadas *deep features* ou *learned features* (Zhang et al., 2023). Atualmente existem alguns tipos de redes neurais artificiais, cada uma com suas peculiaridades quanto à extração de características, sendo as Redes Neurais Convolucionais, *Convolutional Neural Network* (CNN) as mais importantes para este trabalho (Data Science Academy, 2023).

Como os DGA são variados em função da maneira com que são formados, alguns usando caracteres ordenados de maneira pseudo-randômica, outros utilizando apenas parte do alfabeto e ainda outros utilizando listas de palavras, a tarefa de se construir manualmente características que representem todas essas variações se torna relativamente complexa, sob o risco de se produzir características que não sejam úteis ou que colaborem pouco para a identificação de um padrão. Nesse sentido, a abordagem com aprendizado profundo, extraindo *deep features* diretamente dos exemplos fornecidos, se apresenta mais viável e assertiva.

A seguir veremos em maiores detalhes o funcionamento da unidade fundamental das redes neurais artificiais, o neurônio artificial, o mecanismo de ajuste de pesos das redes neurais artificiais, o conceito de redes neurais artificiais profundas e mais detalhes sobre as CNNs.

2.4.1 Redes Neurais Artificiais

Um neurônio artificial é uma representação computacional baseada no comportamento dos neurônios biológicos. Nos neurônios naturais, os sinais são recebidos através de sinapses

localizadas nos dendritos ou na membrana celular. Quando esses sinais atingem uma intensidade suficiente para superar um determinado limiar, o neurônio é ativado e gera um impulso elétrico que percorre o axônio. Esse impulso pode então ser transmitido para outras sinapses, podendo ativar outros neurônios (Gershenson, 2003).

A complexidade dos neurônios biológicos é significativamente simplificada quando criamos modelos de neurônios artificiais. Esses neurônios artificiais são formados essencialmente por entradas, análogas às sinapses, que são multiplicadas por pesos representando a intensidade dos sinais, e processadas por uma função matemática que decide se o neurônio será ativado. Outra função, que pode ser uma simples identidade, calcula a saída do neurônio artificial, às vezes considerando um limiar. As redes neurais artificiais conectam múltiplos neurônios artificiais para processar informações de forma coordenada (Gershenson, 2003).

Figura 7 – Representação básica do neurônio artificial.

Fonte: Elaborado pelo Autor.

Quanto maior for o peso de um neurônio artificial, maior será a influência da entrada associada a ele. Os pesos também podem ser negativos, o que significa que o sinal correspondente é inibido. A operação realizada pelo neurônio artificial depende diretamente dos valores desses pesos que, quando ajustados corretamente, podem obter a saída desejada para determinadas entradas.

No entanto, quando lidamos com uma rede neural artificial composta por centenas ou milhares de neurônios, seria extremamente difícil ajustar manualmente todos os pesos. Felizmente, existem algoritmos que ajustam esses pesos automaticamente, permitindo que a rede produza a saída desejada. Esse processo de ajuste é conhecido como aprendizado ou treinamento. Para a atualização dos pesos durante o processo de treinamento de uma rede neural artificial, o algoritmo mais comumente utilizado é o de *backpropagation*, ou derivações do mesmo, proposto por (Rumelhart; McClelland; Group, 1986).

Existem inúmeros tipos de redes neurais artificiais, cada uma com diferentes aplicações. Desde o modelo neural pioneiro de (McCulloch; Pitts, 1988), centenas de modelos diferentes foram desenvolvidos. As variações entre eles podem envolver as funções utilizadas, os valores

permitidos, a topologia da rede ou os algoritmos de aprendizado, entre outros aspectos. Nesse universo as Redes Neurais Convolucionais são as mais importantes para este trabalho, pois todos os modelos desenvolvidos no decorrer desta pesquisa foram desenvolvidos baseados nesse tipo de rede neural artificial que será melhor abordada na Seção 2.4.2.

2.4.2 Redes Neurais Convolucionais

As Redes Neurais Convolucionais (CNNs) são uma classe de redes neurais artificiais projetadas para processar dados com uma estrutura de grade, como imagens e séries temporais. Inspiradas na organização do córtex visual de animais, as CNNs têm se mostrado altamente eficazes em tarefas de visão computacional, incluindo reconhecimento de padrões, detecção de objetos e segmentação de imagens. Sua arquitetura é composta por camadas convolucionais que permitem a extração automática de características dos dados de entrada, facilitando a identificação de padrões complexos sem a necessidade de intervenção manual (Data Science Academy, 2023).

Este tipo específico de rede neural artificial possui como características essenciais a aplicação dos princípios de invariância e localidade, além de ter na operação de convolução sua operação fundamental no objetivo de extrair características representativas dos dados de entrada (Zhang et al., 2023).

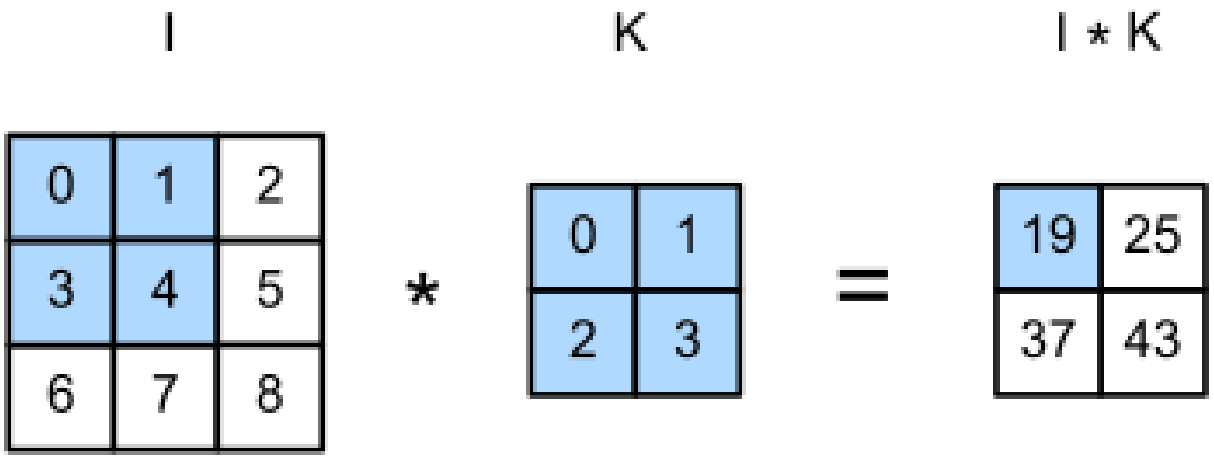
A invariância, especialmente a invariância espacial, é uma propriedade das CNNs que permite à rede reconhecer padrões independentemente de sua posição na entrada. Isso é obtido através do compartilhamento de pesos, onde o mesmo kernel é aplicado em diferentes regiões da entrada, garantindo que a rede responda de maneira consistente a padrões semelhantes, independentemente de sua localização. Além disso, operações como o *pooling* auxiliam na obtenção de invariância a pequenas translações e deformações, tornando a rede mais robusta a variações nos dados (Zhang et al., 2023).

O princípio da localidade nas CNNs refere-se à capacidade da rede de focar em padrões locais presentes nos dados de entrada. Isso é alcançado por meio de campos receptivos locais, onde cada neurônio em uma camada convolucional está conectado apenas a uma região específica da camada anterior. Essa abordagem permite que a rede capture características locais importantes, como bordas e texturas, que são fundamentais para a compreensão de estruturas mais complexas em níveis superiores (Zhang et al., 2023).

Por meio da operação de convolução, as CNNs são capazes de extrair *deep features* de imagens, identificando linhas, bordas, cantos e paleta de cores em áreas específicas (Data Science Academy, 2023). A operação de convolução tem o objetivo de extrair características por meio da aplicação de um kernel de tamanho definido que percorre a imagem aplicando uma função matemática aos valores de intensidade luminosa dos pixels abrangidos pelo kernel (Goodfellow; Bengio; Courville, 2016). A operação de convolução, portanto, é um processo de multiplicação de matrizes. A Figura Figura 8 apresenta um kernel K com valores definidos e o resultado da

aplicação do mesmo sobre uma matriz I gerando a matriz I*K, onde o asterisco (*) representa a operação de convolução.

Figura 8 – Processo de Convolução 2D.



Fonte: Adaptado de (Zhang et al., 2023).

As CNNs são usadas geralmente para aplicações com imagens, como o reconhecimento e classificação de caracteres manuscritos, reconhecimento óptico de caracteres (OCR) ou arquivos de áudio representados na forma de espectrogramas. Este tipo de rede neural artificial tem sido responsável por impulsionar avanços na área de Visão Computacional, mas também tem apresentado bons resultados em outras diversas aplicações (Data Science Academy, 2023).

O exemplo apresentado pela Figura Figura 8 remete à operação de convolução sobre uma estrutura 2D, mas a mesma técnica pode ser aplicada a estruturas mais complexas em 3D como objetos e imagens em três dimensões, ou mais simples de 1D, como vetores de valores sequenciais de um exame de eletrocardiograma, por exemplo. Os DGA são basicamente sequências de caracteres ordenados de forma lógica por um algoritmo, sendo assim um vetor de valores sequenciais, essa característica torna plenamente viável a aplicação de CNN 1D para a extração de *deep features* desses nomes de domínio.

2.4.3 Processamento de Linguagem Natural

O Processamento de Linguagem Natural (PLN) é um conjunto de técnicas computacionais usadas para analisar e representar automaticamente idiomas humanos. No entanto, para que a análise automática de texto alcance o nível dos humanos, as máquinas precisam ter um entendimento muito mais profundo da linguagem natural, algo que ainda está, de certa forma, distante da realidade (Chowdhary, 2020).

Há vários exemplos de aplicação do PLN, como a recuperação de informações online, agregação e respostas a perguntas, os quais geralmente se baseiam em algoritmos que usam representações textuais de páginas da web e, em certa medida, técnicas de PLN. Esses algoritmos são eficazes na recuperação de textos, divisão em partes, verificação de ortografia e análise no

nível de palavras, porém, estes têm dificuldade em realizar uma análise satisfatória no nível de frase e parágrafo. Portanto, quando se trata de interpretar frases e extrair informações significativas, as capacidades desses algoritmos ainda são bastante limitadas(Chowdhary, 2020).

Na linguagem natural as palavras são as unidades básicas e a forma com que são organizadas e aplicadas tem a função de transmitir significados. Os vetores de palavras, como o próprio nome sugere, são representações vetoriais usadas para expressar palavras e também podem ser interpretados como vetores de características ou representações das palavras. O processo de associar palavras a vetores reais é conhecido como incorporação de palavras (word embedding). Nos últimos anos, a prática da incorporação de palavras se estabeleceu progressivamente como um conhecimento fundamental para a área de processamento de linguagem natural (Zhang et al., 2023).

Os nomes de domínio de segundo nível são basicamente sequências de caracteres ordenados com alguma semântica, no caso de domínios legítimos a semântica é humana na maioria das vezes, para os DGA a semântica depende do algoritmo responsável pela geração desse nome de domínio. Partindo desse princípio, este trabalho aborda a incorporação em nível de caracteres, a fim de buscar extrair as relações entre os caracteres presentes em determinado nome de domínio.

2.4.4 Aprendizado Incremental

O aprendizado incremental tem se tornado um grande objeto de estudo para a área de IA moderna (Nagarikar et al., 2023). Comparado ao ML clássico, onde o algoritmo é treinado com uma grande massa de exemplos ao mesmo tempo, no aprendizado incremental o algoritmo deve ser capaz de aprender incrementalmente, recebendo novos exemplos de treinamento, sem perder o que havia aprendido anteriormente (Yang; Gu; Wu, 2019), algo mais parecido com o modo de aprendizagem dos seres humanos. Para o aprendizado incremental existem três tipos ou cenários fundamentais: incremento de tarefa, incremento de domínio e incremento de classe (Ven; Tuytelaars; Tolias, 2022).

No incremento de tarefa o algoritmo deve ser capaz de adicionar tarefas distintas (Ven; Tuytelaars; Tolias, 2022), como, por exemplo, um algoritmo que faz a tradução de textos de português para inglês, deve passar a fazer a transcrição de áudios em português para inglês, ambas as tarefas são tarefas de tradução, porém são tarefas distintas, nesse caso o algoritmo pode utilizar seu conhecimento prévio sobre ambos os idiomas e agora executar uma nova tarefa.

No incremento de domínio, semelhantemente ao incremento de tarefas, o algoritmo deve ser capaz de aprender incrementalmente um novo domínio da mesma tarefa que este já executa (Ven; Tuytelaars; Tolias, 2022). Um exemplo seria um algoritmo que faz a tradução de textos de português para inglês e deve passar a fazer a tradução de textos de português para espanhol. Sendo a tarefa de tradução de texto a mesma, porém uma nova saída será adicionada para a tradução em um novo idioma.

O incremento de classe por sua vez está ligado diretamente a algoritmos classificadores, neste caso o algoritmo de IA deve conseguir adicionar novas classes ao seu classificador conforme

estas forem percebidas (Ven; Tuytelaars; Tolias, 2022). Um exemplo de incremento de classe seria um algoritmo classificador que classifica imagens de cães e gatos e agora precisa adicionar girafas à sua classificação.

Ocorre que, em redes neurais artificiais, um problema importante quando se fala em aprendizado incremental é que, redes neurais artificiais, principalmente redes recorrentes, têm a tendência de "esquecer" rapidamente os parâmetros aprendidos quando recebem um conjunto completamente novo de exemplos de treinamento, este fenômeno é denominado esquecimento catastrófico (Ramesh; Chaudhari, 2022).

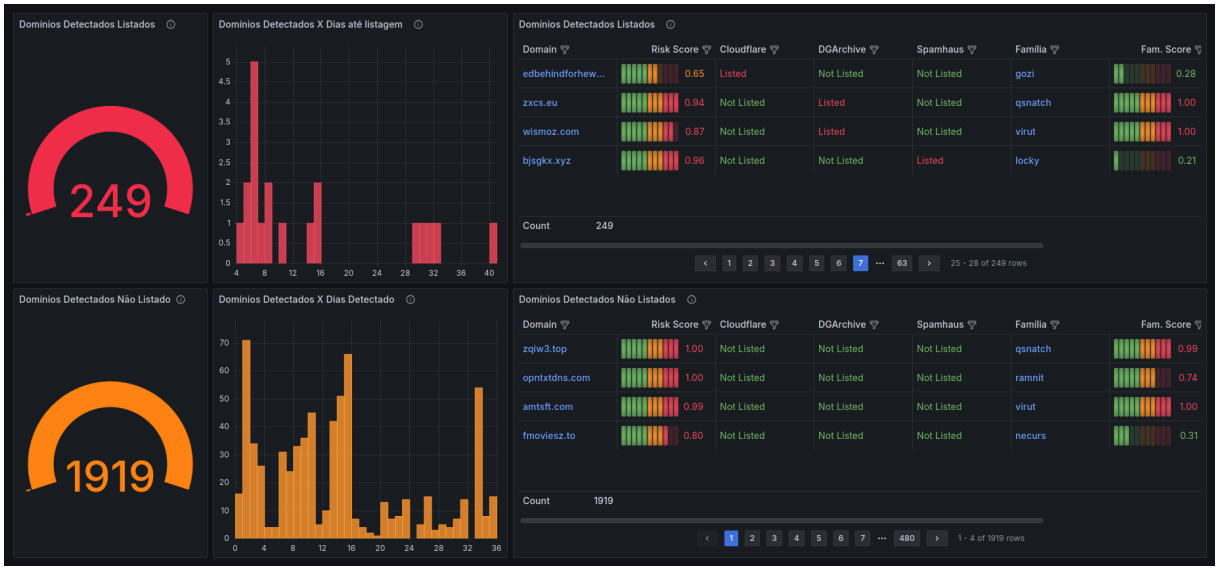
Dentre as abordagens recentes no sentido de evitar o esquecimento catastrófico no treinamento incremental de redes neurais, a abordagem utilizando transferência de aprendizado (Llopis-Ibor et al., 2023) é a sugerida pelo desenvolvedor da biblioteca para desenvolvimento de redes neurais artificiais profundas Tensorflow (Abadi et al., 2015).

A aplicação da técnica de aprendizado incremental pro transferência de aprendizado neste trabalho busca apresentar um mecanismo viável para a manutenção e evolução de performance do modelo de aprendizado profundo desenvolvido, apresentando novos exemplos de DGA ao modelo detector conforme estes surgem no ambiente, sem que seja necessário o treinamento do modelo com todos os exemplos existentes até aquele momento e mantendo os níveis de performance do modelo elevados na realização da tarefa proposta.

2.5 PAINEL GRAFANA

Grafana (Grafana Labs, 2023) é uma ferramenta de código aberto para visualização de dados capaz de se conectar a uma lista diversa de tipos de bases de dados visando gerar gráficos, listas e painéis. A utilização de painéis dinâmicos para a visualização de dados é amplamente utilizada para entregar a equipes especialistas informações mais claras e acessíveis para a tomada de decisão. Com uma vasta diversidade de gráficos disponíveis, esta ferramenta permite exibir grandes volumes de dados de forma clara e objetiva, enriquecendo a informação acessível. A Figura Figura 9 demonstra um painel Grafana produzido em testes piloto para este trabalho.

Figura 9 – Painel Grafana em testes piloto.



Fonte: Elaborado pelo autor.

A partir das coletas de consultas DNS com a ferramenta de DNS passivo, as consultas tratadas e apresentadas ao modelo de aprendizado profundo treinado podemos produzir painéis para visualização dos resultados. O painel de exemplo apresentado na Figura Figura 9 exibe o total de domínios maliciosos detectados listados em alguma das listas de bloqueio utilizadas (Cloudflare, DGArchive e DBL Spamhaus), total de domínios detectados não listados em listas de bloqueio, a distribuição dos mesmos por tempo e tabelas com os domínios maliciosos detectados, seu *Risk Score* atribuído pelo modelo de aprendizado profundo e seu estado nas três listas onde foi consultado.

2.6 TRABALHOS RELACIONADOS

Dada a amplitude do problema de detecção automática de domínios gerados por algorítmos, atualmente muitos trabalhos têm sido desenvolvidos com este objetivo, lançando mão das mais diversas abordagens e técnicas de inteligência artificial.

Em (Sun; Liu, 2023) os pesquisadores propõem uma abordagem utilizando características desenvolvidas à mão, algumas baseadas em características léxicas do nome de domínio e outras baseadas em dados whois. Ao todo são sete características extraídas do nome de domínio e outras dezesseis de dados whois. Neste trabalho os autores usaram a base Alexa 1 Million (Sun; Liu, 2023) para exemplos de domínios legítimos e o feed NetLab360 (NetLab 360, 2022) para exemplos de 58 famílias de DGAs. Essas características desenvolvidas manualmente são fornecidas como entrada para redes neurais profundas, os autores propõem a utilização de um modelo de aprendizado profundo híbrido de redes BiLSTM, mecanismos de atenção e uma parte convolucional. Os resultados obtidos pelo modelo são muito bons, com acurácia de 97,13%, Precisão de 96,27%, Recall de 97,65% e F1 Score de 96,96%.

Há trabalhos com abordagens de aprendizado profundo e tokenizadores pré-treinados (Gogoi; Ahmed, 2023; Huang et al., 2022), chamados transformers, utilizados principalmente no processamento de linguagem natural (NLP), estes fazem a tokenização dos caracteres dos nomes de domínio de maneira sensível ao contexto, não somente transpondo uma palavra ou caractere diretamente para uma representação numérica, significa dizer que um caractere ou palavra pode ter valores distintos dependendo das palavras ou caracteres próximos.

No trabalho (Gogoi; Ahmed, 2023) os pesquisadores utilizam um modelo transformer pré-treinado chamado CANINE, proposto em (Clark et al., 2022), para tokenização dos nomes de domínio e uma rede totalmente conectada (DNN) para fazer a classificação. O trabalho se apoia na base Alexa 1 Million para exemplos de domínios legítimos e NetLab360 para exemplos de domínios gerados por algorítmo. Seus resultados são excelentes, com Acurácia de 99%, Precisão de 99%, Recall de 99% e F1-Score de 99%.

Uma abordagem semelhante é proposta por (Huang et al., 2022) que lança mão do transformer pré-treinado BERT (Devlin et al., 2018) para tokenização dos nomes de domínio, mas também utiliza camadas convolucionais em seu modelo. Os pesquisadores neste trabalho utilizaram a base Alexa 1 Million para exemplos de nomes de domínio legítimos e as bases NetlLab360, Bambenek Colsulting e DGArchive para exemplos de domínios gerados por algoritmo. Com uma base mais ampla e variada, o modelo obteve resultados muito bons, com Acurácia de 96,08%, Precisão de 95,80%, Recall de 96,20% e F1-Score de 96% para a detecção dos DGAs. O mesmo trabalho apresenta um modelo multiclasse para a classificação dos exemplos DGA em famílias, e neste segundo caso o modelo performou com Acurácia 81,20%, Precisão de 80,46% e F1-Score de 80,97%.

Já (Tuan; Long; Taniar, 2022) apresenta dois modelos baseados em redes LSTM e mecanismos de atenção na fase de codificação dos nomes de domínio, um para detectar domínios gerados por algoritmos e outro para classificar estes domínios por família de ameaça. Este trabalho utiliza a base Alexa 1 Million para exemplos de domínios legítimos, e NetLab360, Andrey Abakumov's DGA, Bambenek Consulting e UMUDGA para exemplos de domínios gerados por algoritmo e sua performance descrita pelos autores foi: Acurácia de 99%, Precisão de 99%, Recall de 99% e F1-Score de 99% para a detecção média dos DGAs em todos os datasets. Para a classificação multiclasse por famílias, as métricas apresentadas no trabalho ficaram em Acurácia de 86%, Precisão de 87%, Recall de 86% e F1-Score de 85%.

Outro trabalho que aborda a detecção de DGAs por meio de modelos complexos de aprendizado profundo é (Highnam et al., 2020) onde os pesquisadores propõem um modelo complexo denominado Bilbo, a partir da concatenação de redes LSTM, CNN e Densas, e compara seus resultados com redes de camada única CNN, LSTM e DNN. As métricas apresentadas pelos autores no paper são: acurácia de 96,56%, precisão de 95,57%, recall de 97,66% e f1-score de 96,60%. Este trabalho disponibiliza o código-fonte de seu modelo em repositório público, o que nos permitiu sua reprodução e comparação direta com os modelos propostos desenvolvidos durante esta pesquisa. Nesta mesma linha há o trabalho (Ravi et al., 2023) que apresenta a

aplicação de redes neurais simples CNN, LSTM e BiLSTM para a detecção e classificação de DGAs. Utilizando um conjunto de dados com 21 famílias, as melhores métricas para classificação binária apresentadas pelo trabalho foram para a arquitetura B-GRU com Precisão de 92%, Recall de 92% e F1 Score de 93%.

Uma abordagem diferente é apresentada por (Park et al., 2022), com uma proposta de detecção de anomalias por meio de um autoencoder[2] convolucional. Os dados de entrada são compostos por doze características léxicas extraídas dos nomes de domínio. O modelo é treinado exclusivamente com exemplos legítimos e um limite de erro é definido. Quando exemplos de domínios maliciosos são apresentados à rede neural, esta tende a reconstruir os dados com uma taxa de erro maior. Reconstruções acima do limite de erro definido são consideradas como positivas para DGA. Como exemplos de domínios legítimos, os pesquisadores utilizaram a base Alexa Top 1 Million e a base AmritaDGA e Bambenek Consulting para exemplos de domínios gerados por algoritmos. Os resultados apresentados no trabalho foram de acurácia 99,8%, precisão 100%, recall 99% e f1-score 99,49%.

Percebe-se, portanto, que há algum espaço para melhorias na detecção dos domínios gerados por algoritmo, principalmente quando pensamos no surgimento de novas ameaças e na evolução dos algoritmos geradores de domínio, sendo necessário que o modelo consiga acompanhar estas mudanças no ambiente e evoluir gradativamente sem que haja perda em sua capacidade de detecção. Da mesma forma, todos os trabalhos se utilizam de bases bem delimitadas para treinamento, validação e testes de seus modelos, o que pode limitar sua utilização em aplicações de mundo real, onde novas ameaças surgem diariamente e precisam ser detectadas. Por esta razão, nossa proposta busca, não só detectar os domínios gerados por algoritmos mas também aplicar a técnica de aprendizado incremental por transferência de aprendizado a fim de manter o modelo atualizado à medida que novos exemplos e famílias surgem, além de buscar a aplicação do modelo desenvolvido em ambiente de mundo real com o uso de DNS Passivo e a apresentação dos resultados com painéis Grafana como uma ferramenta de inteligência para especialistas em cibersegurança.

[2] Um autoencoder é uma estrutura de rede neural que busca reconstruir em sua saída os dados de entrada com o menor erro possível (Goodfellow; Bengio; Courville, 2016)

3 METODOLOGIA

Neste capítulo abordaremos as etapas e procedimentos adotados para o desenvolvimento do trabalho. Serão apresentados os pontos mais importantes do projeto proposto, suas fontes de dados, o pré-processamento aplicado aos dados originais, os modelos aplicáveis ao objeto de pesquisa e as métricas que deverão ser aplicadas em cada etapa a fim de verificar a performance do modelo proposto.

3.1 TECNOLOGIAS E MATERIAIS

Para o desenvolvimento deste trabalho foram utilizados um notebook com 32GB de memória RAM, processador Intel Core i5 de 11ª Geração com 12 núcleos de processamento, GPU RTX 3050 com 4GB de memória de processamento gráfico, 1TB de armazenamento em unidades NVMe e 4TB de armazenamento em disco rígido externo para o tratamento dos dados, construção dos algoritmos de treinamento dos modelos e análise dos resultados.

Um segundo computador com 16GB de memória RAM, dois processadores Intel Xeon com 16 núcleos de processamento cada e 900 GB de armazenamento em unidades SAS foi utilizado para armazenar as coletas de DNS passivo provenientes do servidor DNS recursivo da Reitoria Unesp no campus da Unesp FCL Assis, bem como hospedar o sistema Grafana para a apresentação dos painéis de monitoramento DNS.

Todos os algoritmos foram desenvolvidos utilizando a linguagem de programação Python 3.8 com o auxílio das bibliotecas Pandas e Numpy para manipulação dos conjuntos de dados. A biblioteca scikit-learn foi utilizada para a apresentação das métricas dos modelos e em etapas do tratamento dos conjuntos de dados.

Os modelos de aprendizado profundo foram desenvolvidos utilizando a biblioteca de código aberto Tensorflow, em sua versão 2.13.0, desenvolvida e mantida pelo Google. Todos os códigos desenvolvidos para a execução dos experimentos relacionados a este trabalho foram publicados na plataforma Github e estão disponíveis em https://github.com/rafagregs/dga-detection.

3.1.1 Fontes de dados

Para o treinamento e teste dos modelos propostos neste trabalho foi utilizada como fonte de exemplos de domínios legítimos a base Majestic Million, que engloba os 1 milhão de domínios mais acessados na internet; essa lista se atualiza diariamente e está disponível para download sem nenhum tipo de restrição.

Como fonte de exemplos de domínios gerados por algoritmo utilizamos a base DGArchive mantida pelo Instituto Fraunhofer de Comunicação, Processamento de Informação e Ergonomia FKIE, instituição alemã focada em processos de desenvolvimento de tecnologias para detecção e mitigação precoce de riscos, que gentilmente nos forneceu um acesso à sua API para consulta

tanto das bases históricas de DGAs quanto para os exemplos catalogados diariamente. Esta base foi escolhida por englobar feeds de DGAs provenientes de diversas origens, como o feed fornecido pela Bambenek Consulting e o feed fornecido pela NetLab360, outras duas bases muito utilizadas separadamente em trabalhos acadêmicos, o que torna a base DGArchive mais complexa e heterogênea, sendo, portanto, mais próxima do ambiente de mundo real.

Adicionalmente, uma terceira fonte de dados, desta vez não rotulada, foi construída com a utilização do serviço de DNS passivo, explicado em detalhes na seção 2.1.3, implantado no servidor DNS recursivo da Reitoria Unesp e mantido no datacenter da Unesp FCL-Assis, com a coleta e armazenamento em banco de dados das consultas DNS realizadas por estações de trabalho do ambiente administrativo, laboratórios e dispositivos conectados à rede wi-fi de toda a universidade.

3.1.2 Pré-processamento

Uma etapa simples, porém muito importante é a etapa de pré-processamento dos dados, seu objetivo é construir o vetor de entrada para o modelo proposto. Como o único dado utilizado como entrada no modelo proposto é o próprio nome de domínio a ser classificado, efetuamos duas etapas de pré-processamento.

Primeiramente, convertemos cada caractere presente no nome de domínio para seu respectivo valor ASCII, a fim de obter a representação numérica do nome de domínio.

Após a conversão dos caracteres de cada domínio para ASCII o resultados são vetores com comprimentos distintos, uma vez que os nomes de domínio não possuem um comprimento definido, porém, para que uma sequência de exemplos seja entregue a uma rede neural, estes exemplos precisam possuir as mesmas dimensões, chamado de *input shape*.

Uma das técnicas mais simples e utilizadas para se ajustar a dimensão de exemplos de entrada com dimensões distintas é o preenchimento com zeros ou *zero padding*, onde se define a dimensão a ser utilizada e, quando o exemplo possuir dimensões inferiores às esperadas, este é preenchido com zeros (Najeh; Lohr; Leduc, 2023; Maniriho; Mahmood; Chowdhury, 2023; Wei et al., 2024).

Em nosso conjunto de dados inicial, verificamos que apenas 0,0056% das amostras possuíam mais de 70 caracteres, portanto, o *input_shape* foi definido em 70 valores ASCII, domínios com menos caracteres foram preenchidos com zero a partir do final.

Para os nomes de domínio mais longos que 70 caracteres definimos que seriam codificados até o limite de 70, desprezando-se os caracteres excedentes. Dessa forma, cada nome de domínio se torna um vetor de entrada com dimensão 1x70 e está pronto para ser apresentado como entrada ao modelo proposto.

Naturalmente, quando efetuamos o preenchimento de domínios mais curtos com zero ou desprezamos parte de domínios mais longos estamos modificando estes nomes de domínio e adicionando certa poluição aos dados, porém, a partir dos resultados obtidos e apresentados

neste trabalho, podemos afirmar que o impacto deste processo não impactou significativamente na capacidade do modelo em detectar os nomes de domínios maliciosos.

A etapa de pré-processamento dos exemplos obtidos da lista Majestic Million e do feed fornecido pela API DGArchive consiste em transpor cada caractere que compõe o nome de domínio para seu respectivo código ASCII até o limite de 70 caracteres, exemplos com comprimento maior que 70 caracteres foram cortados, exemplos com menos de 70 caracteres foram completados com zeros a partir do final, gerando assim para cada exemplo um vetor inicial de 70 valores ACII ordenados.

Para os exemplos coletados em ambiente real com a ferramenta de DNS passivo primeiramente separamos somente TLD e nome de domínio de segundo nível, como google.com, portanto todas as consultas por nomes totalmente qualificados como www.google.com são tratados para estarem no formato esperado. Depois o processo de obtenção do vetor de 70 valores ASCII segue o modelo aplicado aos exemplos obtidos pelas bases mencionadas anteriormente. A Tabela 3 apresenta exemplos de nomes de domínio, seus rótulos e seus vetores ASCII.

Tabela 3 – Exemplos de nomes de domínio, família, rótulo e seus caracteres codificados para ASCII.

domínio	família	rótulo	1	2	3	4	5	6	7	...	70
epyuoxblj.com	abcbot	1	101	112	121	117	111	120	98	...	0
6a8ddedf.org	antavmu	1	54	97	56	100	100	101	100	...	0
tbrbmen.com	banjori	1	116	98	114	98	109	101	110	...	0
...	...	...	...	...	...	...	...	...	...	...	...
weightrainer.net	legit	0	119	101	105	103	104	116	114	...	0
deltasan.ru	legit	0	100	101	108	116	97	115	97	...	0
bagiran.ir	legit	0	98	97	103	105	114	97	110	...	0

Fonte: Elaborada pelo autor.

Os vetores de valores ASCII, produzidos durante o pré-processamento são utilizados como entrada para o modelo de aprendizado profundo proposto neste trabalho e que serão melhor detalhados na Seção 3.2 a seguir, bem como para os demais modelos obtidos em trabalhos recentes utilizados para comparação direta de resultados.

3.1.3 Métricas e Avaliações

Para a avaliação do modelo proposto e desenvolvido neste trabalho, em todas as etapas de testes, verificamos as métricas mais aceitas e utilizadas atualmente para a avaliação de desempenho de modelos de inteligência artificial, a saber Precisão 1, Recall 2, F1-Score 3, Acurácia 4 e a Taxa de Falsos Positivos 5, obtidas conforme a seguir:

$$Precision = \frac{TP}{FP + TP} \tag{1}$$

$$Recall = \frac{TP}{FN + TP} \tag{2}$$

$$F1 - Score = 2 \times \frac{Recall \times Precision}{Recall + Precision} \tag{3}$$

$$Accuracy = \frac{TN + TP}{TN + TP + FN + FP} \tag{4}$$

$$FalsePositiveRate(TFP) = \frac{FP}{TN + FP} \tag{5}$$

Onde:

- Verdadeiro Positivo (TP): Domínio DGA classificado como DGA;
- Verdadeiro Negativo (TN): Domínio Legítimo classificado como Legítimo;
- Falso Positivo (FP): Domínio Legítimo classificado como DGA;
- Falso Negativo (FN): Domínio DGA classificado como Legítimo.

Os resultados das métricas apresentadas acima foram tabulados conforme a especificidade da análise necessária, alguns gráficos foram construídos utilizando os valores obtidos nos testes para as métricas mais relevantes, a fim de facilitar a compreensão e discussão dos resultados. Estes estão apresentados no Capítulo 4.

3.2 MODELO DE APRENDIZADO PROFUNDO

Esta seção apresenta o modelo de aprendizado profundo proposto neste trabalho, sua estrutura de camadas, funções e hiperparâmetros. Também é apresentado o processo de treinamento incremental por transferência de aprendizado proposto para a manutenção e evolução das métricas do modelo.

3.2.1 Modelo Detector

O modelo detector proposto neste trabalho tem o objetivo de classificar um nome de domínio fornecido na entrada do modelo entre legítimo ou DGA, o mesmo foi construído baseado em uma CNN, tendo como entrada os nomes de domínio convertidos, caractere a caractere até o limite de 70 caracteres, para seus respectivos códigos ASCII como demonstrado na Seção 3.1.2.

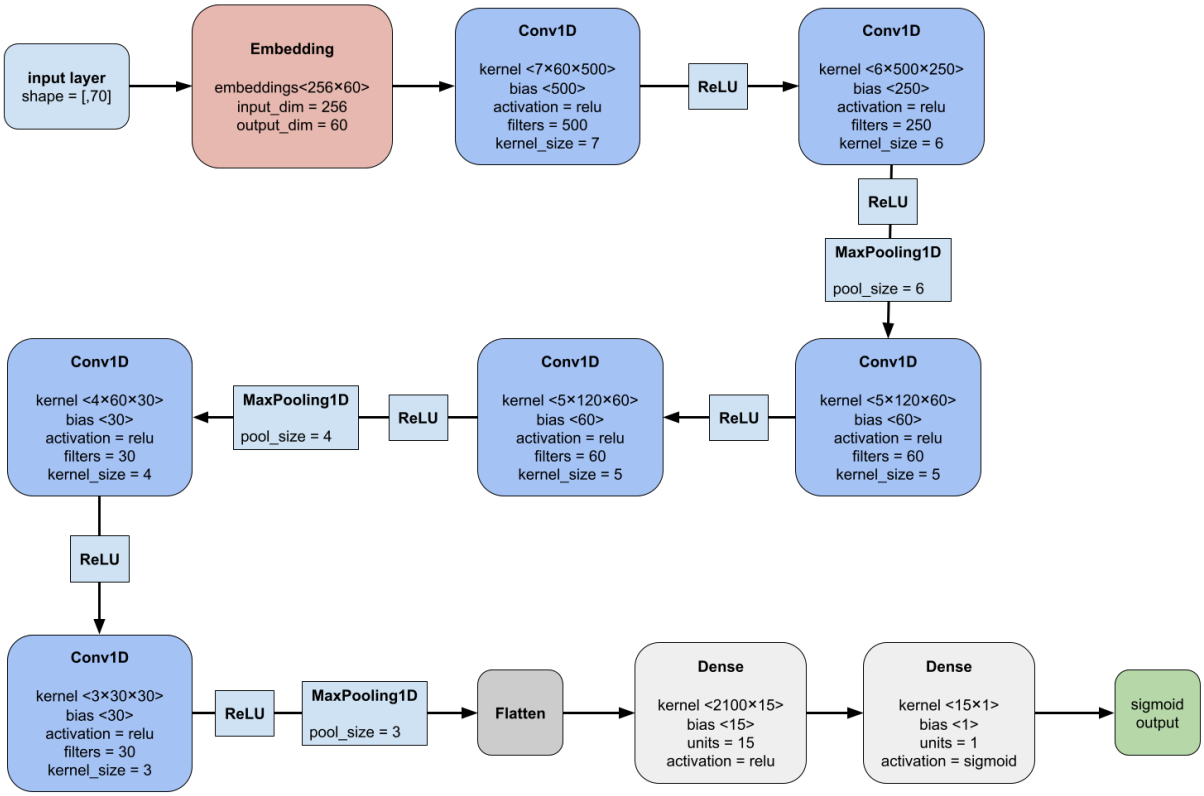
Este vetor de características inicial de dimensão 1x70 é então passado através de uma camada embedding que converte cada valor ASCII em um vetor de 60 características relacionadas ao contexto do caractere no nome de domínio.

A saída da camada embedding é conectada a uma CNN de seis camadas, que visa extrair *deep features* relevantes dos nomes de domínio a partir dos vetores recebidos da camada embedding imediatamente superior.

Além disso, a função relu é aplicada entre cada camada convolucional para favorecer a dispersão da rede neural, retornando zero para quaisquer valores não positivos, também aplicamos a operação maxpooling a cada duas camadas convolucionais a fim de destacar características

específicas das classes analisadas. A Figura Figura 10 apresenta uma visão geral do modelo detector.

Figura 10 – Visão conceitual do modelo detector proposto.



Fonte: Elaborado pelo autor.

O neurônio único na última camada da rede neural profunda utiliza a função de ativação sigmoid, que retorna um valor entre 0 e 1, sendo considerados positivos para DGA todos os valores maiores ou iguais a 0,5 e negativos, portanto legítimos, todos os valores menores que 0,5.

3.2.1.1 Camada Embedding

A camada embedding é uma camada utilizada em redes neurais profundas, especialmente em PLN e problemas que envolvam dados categóricos. Ela tem a função de transformar representações discretas (como palavras ou categorias) em vetores densos e contínuos em um espaço de menor dimensão (Goodfellow; Bengio; Courville, 2016; Zhang et al., 2023), facilitando a captura de relações semânticas entre as categorias. Camadas embedding são treináveis, portanto a representação de cada palavra pode ser aprendida pela rede neural em função do seu contexto.

Em uma camada embedding, dois parâmetros são importantes para o correto funcionamento da mesma, *input_dim* e *output_dim*. O parâmetro *input_dim* especifica o tamanho do vocabulário ou o número total de categorias distintas, enquanto *output_dim* define o tamanho dos vetores

densos resultantes. Exemplificando, uma camada de embedding com *input_dim*=1000 e *output_dim*=60, indica um vocabulário de 1000 palavras, cada uma mapeada para um vetor de dimensão 60.

Enquanto o valor do *input_dim* é determinado pela quantidade de palavras diferentes existentes no vocabulário utilizado em cada experimento e pode facilmente ser determinado, a escolha do valor de *output_dim* não é bem definida na literatura e varia conforme a aplicação. Em tarefas de processamento de linguagem natural, dimensões entre 100 e 300 são frequentemente utilizadas para representar palavras em trabalhos de análise de sentimentos (Trisna et al., 2024; Anil; Kaur, 2024; Mussiraliyeva et al., 2024).

Neste trabalho utilizamos embedding em nível de caracteres, pois como não possuímos frases, mas sim nomes de domínio, onde cada caractere tem sua semântica conforme seu posicionamento no nome de domínio, sendo assim, cada um dos caracteres dos nomes de domínio analisados é apresentado à camada embedding para que seja gerado seu vetor embedding. A camada embedding proposta possui *input_dim*=256, pois este é o tamanho máximo do vocabulário (caracteres distintos) na tabela ASCII de 8 bits.

Como não encontramos consenso na literatura quanto ao tamanho ideal do *output_dim* e nem uma relação direta de proporcionalidade entre este e outros parâmetros da camada embedding, definimos o *output_dim*=60 levando em consideração o comprimento máximo dos nomes de domínio utilizados no experimento inicial, de 70 caracteres, para que o vetor embedding não fosse maior que o próprio nome de domínio. Também levamos em consideração o tamanho total do vocabulário, para que o vetor embedding não fosse tão grande quanto o próprio vocabulário.

3.2.1.2 Rede Neural Convolucional

Como visto na seção 2.4.1, as redes neurais convolucionais são capazes de encontrar características específicas em imagens, exigindo um pré-processamento mínimo (Goodfellow; Bengio; Courville, 2016; Zhang et al., 2023). Neste trabalho, a parte CNN do modelo proposto é projetada para extrair características relevantes das classes analisadas, legítimas e DGAs, a partir do vetor de características gerado pela camada embedding posicionada na entrada de dados do modelo. Nosso modelo possui seis camadas convolucionais de dimensões distintas, formando uma CNN profunda. As dimensões de cada camada convolucional e das demais camadas do modelo proposto podem ser vistas em detalhes na Figura Figura 11.

3.2.1.3 Função ReLU e operação de Maxpooling

Posicionadas entre as camadas convolucionais estão etapas com a função relu e a cada duas camadas convolucionais efetuamos a operação de maxpooling, contribuindo para maximizar a capacidade intrínseca do modelo de discernir e identificar atributos pertinentes associados a cada classe distinta (Nayef Siti Norul Huda Sheikh Abdullah, 2023).

A unidade linear retificada (ReLU) é uma função de ativação que, na prática, transforma toda entrada negativa em zero (Zhang et al., 2023), conforme descrito pela equação 6. Este recurso é desejável por dois motivos: primeiro, os neurônios da rede que recebem valores zero não serão ativados, tornando a rede esparsa.

Em segundo lugar, os valores positivos que serão tratados pela rede tendem a ser mais representativos; assim, a função relu acaba funcionando como um alto contraste, destacando áreas importantes.

$$ReLU(x) = (x)^{+} = max(0, x) \tag{6}$$

As operações de maxpooling reduzem a dimensão do mapa de características extraído pelas camadas convolucionais (Zhang et al., 2023). No modelo proposto neste trabalho, as operações de maxpooling são utilizadas para que os mapas de características destacados pela função relu possam ser condensados. Esta integração de camadas CNN, função relu e operações de maxpooling pretende aumentar a capacidade do modelo em extrair e processar informação de forma eficiente, facilitando assim o seu desempenho na identificação de características relevantes para a classificação dos nomes de domínio.

3.2.1.4 Camadas Flatten e Densas

A camada flatten foi utilizada após a parte CNN do modelo proposto para remodelar o formato do vetor de características produzido pelas camadas superiores, tornando-o compatível com as camadas densas totalmente conectadas posicionadas no final do modelo. Finalmente, duas camadas totalmente conectadas, com 15 e 1 neurônio respectivamente, atuam como classificadores usando o vetor de características recebido da camada flatten. A figura Figura 11 mostra o sumário do modelo, suas camadas e suas dimensões.

Figura 11 – Sumário do modelo proposto, todas as camadas e suas dimensões.

```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 embedding (Embedding)       (None, 70, 60)            15360

 conv1d (Conv1D)             (None, 70, 500)           210500

 re_lu (ReLU)                (None, 70, 500)           0

 conv1d_1 (Conv1D)           (None, 70, 250)           750250

 re_lu_1 (ReLU)              (None, 70, 250)           0

 max_pooling1d (MaxPooling1  (None, 70, 250)           0
 D)

 conv1d_2 (Conv1D)           (None, 70, 120)           150120

 re_lu_2 (ReLU)              (None, 70, 120)           0

 conv1d_3 (Conv1D)           (None, 70, 60)            36060

 re_lu_3 (ReLU)              (None, 70, 60)            0

 max_pooling1d_1 (MaxPoolin  (None, 70, 60)            0
 g1D)

 conv1d_4 (Conv1D)           (None, 70, 30)            7230

 re_lu_4 (ReLU)              (None, 70, 30)            0

 conv1d_5 (Conv1D)           (None, 70, 30)            2730

 re_lu_5 (ReLU)              (None, 70, 30)            0

 max_pooling1d_2 (MaxPoolin  (None, 70, 30)            0
 g1D)

 flatten (Flatten)           (None, 2100)              0

 dense (Dense)               (None, 15)                31515

 dense_1 (Dense)             (None, 1)                 16

=================================================================
Total params: 1203781 (4.59 MB)
Trainable params: 1203781 (4.59 MB)
Non-trainable params: 0 (0.00 Byte)
```

Fonte: Elaborado pelo autor.

3.2.1.5 Hiperparâmetros

Outro aspecto importante em modelos de aprendizado profundo são os hiperparâmetros. Em modelos de aprendizagem profunda, hiperparâmetros são parâmetros que não são aprendidos durante o treinamento do modelo, mas precisam ser definidos antes do início deste processo (Koutsoukas et al., 2017; Dalli, 2022). Eles são definidos externamente ao modelo e desempenham um papel crucial na arquitetura e no treinamento da rede neural.

Ao contrário dos parâmetros do modelo, como pesos e vieses, que são ajustados automaticamente pelo algoritmo de otimização durante o treinamento, os hiperparâmetros são predeterminados e impactam diretamente o desempenho e o comportamento do modelo. Assim, os hiperparâmetros utilizados para compilar o modelo proposto são apresentados na Tabela 4.

Tabela 4 – Hiperparâmetros usados para compilar o modelo proposto.

Hiperparâmetro	Valor
Regularizador L1	1e-5
Regularizador L2	1e-4
Otimizador	Adam
Função de Ativação	ReLU
Taxa de Aprendizagem	1e-4

Fonte: Elaborada pelo autor.

Os regularizadores L1 e L2 foram utilizados para aplicar penalidades aos parâmetros das camadas, sendo adicionados à função de perda para evitar *overfitting* (Salehin; Kang, 2023). Os baixos valores adotados, 1e-5 e 1e-4 respectivamente, são baixos o suficiente para não causar impacto repentino no processo de treinamento.

O otimizador Adam foi selecionado porque é um método de descida de gradiente estocástico computacionalmente eficiente para problemas com muitos parâmetros e dados (Kingma; Ba, 2017).

A função de ativação ReLU foi escolhida para aumentar a dispersão da rede devido à sua característica de transformar cada valor negativo recebido em zero (Zhang et al., 2023) e a taxa de aprendizagem do modelo foi definida para um valor baixo, 1e-4, visando uma evolução mais lenta das atualizações dos parâmetros e assim buscar a melhor perda mínima global.

3.2.1.6 Execução do Experimento

O experimento foi realizado em duas etapas, a primeira consistiu em uma avaliação da performance do modelo proposto sobre um conjunto fixo de dados formado por 15 dias de feed DGA obtido pela API DGArchive, entre 22 de julho e 05 de agosto de 2023, composto por cerca de 1,3 milhões de domínios DGA únicos e o conjunto de 1 milhão de domínios legítimos obtidos junto à Majestic Million.

O conjunto de dados foi separado em 70% para treinamento e validação e 30% para teste em três sorteios distintos, as métricas obtidas e comparadas com as métricas de outros modelos. Os modelos de comparação e as métricas obtidas são apresentados na Seção 4.1.

A segunda etapa, mais complexa, onde verificamos a performance do modelo em treinamento incremental foi executada da seguinte forma:

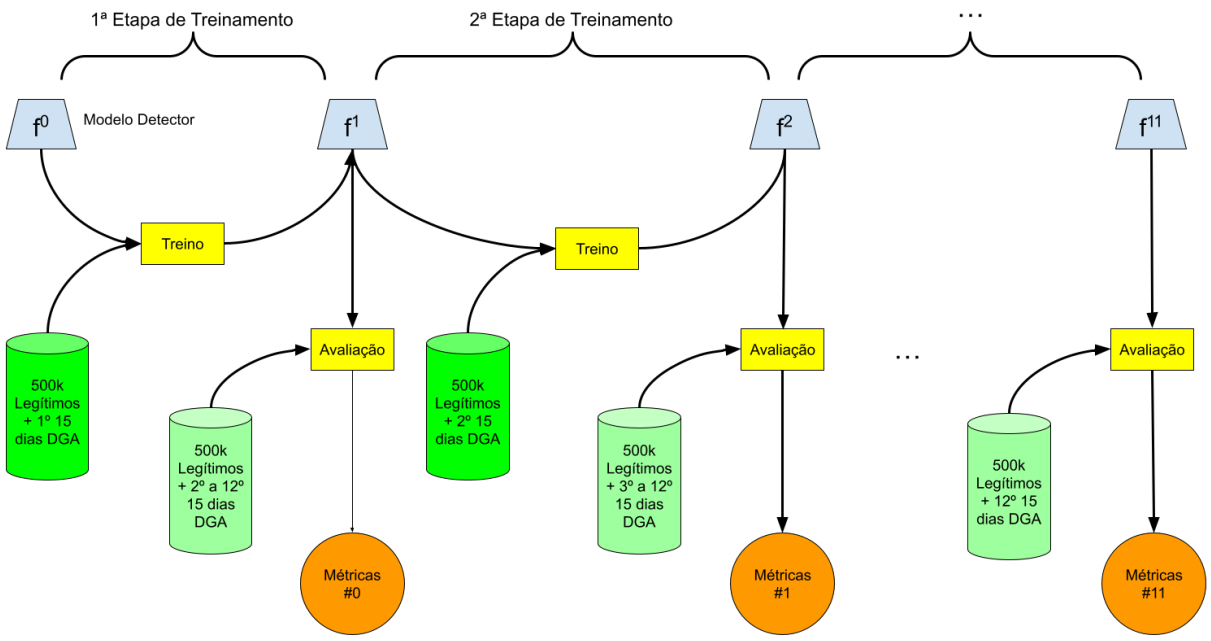
1. Divisão do conjunto de dados legítimos: Utilizou-se o conjunto de dados Majestic Million como referência para domínios legítimos. Esse conjunto foi dividido em duas partes:
    - 500.000 exemplos para treinamento
    - 500.000 exemplos para teste
2. Coleta dos exemplos de domínios DGA: Os exemplos de domínios gerados por algoritmos (DGA) foram obtidos a partir do feed da API do DGArchive. A coleta abrangeu 12

períodos consecutivos de 15 dias, no intervalo de 06 de agosto de 2023 a 01 de fevereiro de 2024. Cada período de 15 dias contém uma quantidade variável de exemplos DGA, sendo em média 1,2 milhões de exemplos em cada período.

3. Execução do treinamento: Para cada uma das 12 etapas de treinamento:
   - Foram utilizados os 500.000 exemplos legítimos reservados para treinamento.
   - Esses exemplos foram combinados com os dados do feed DGA correspondentes ao período de 15 dias em questão.
   - O treinamento do modelo foi realizado por 10 épocas.
4. Predição e avaliação: Após o treinamento, foi realizada a predição utilizando:
   - Os 500.000 exemplos legítimos reservados para teste.
   - Cada um dos conjuntos de domínios DGA dos períodos seguintes de 15 dias.
5. Cálculo das métricas: As métricas de desempenho foram calculadas para cada conjunto de testes, permitindo avaliar a evolução do modelo ao longo do tempo.

A Figura Figura 12 apresenta o processo de treino e avaliação de resultados do modelo de maneira incremental pelo período de 180 dias.

Figura 12 – Processo de treinamento e avaliação incremental do modelo detector.



Fonte: Elaborado pelo autor.

Em redes neurais artificiais, o treinamento incremental sofre um fenômeno chamado esquecimento catastrófico, já mencionado na Seção 2.4.4. Esse fenômeno ocorre quando re-treinamos

um modelo com um conjunto de dados muito diferente do qual o mesmo havia sido treinado anteriormente.

Em nosso experimento, como as classes do modelo não são alteradas por se tratar de um classificador binário, o único cuidado que tomamos é de que, no conjunto de treinamento incremental a cada etapa, não existam apenas exemplos novos para cada classe, mas também exemplos já apresentados ao modelo em momentos de treinamento anteriores.

Como naturalmente uma parte dos domínios DGA são percebidos nos ambientes de rede por períodos maiores do que 15 dias, há, portanto, uma intersecção natural de exemplos entre cada conjunto de treinamento, o que diminuiu a dificuldade do incremento de aprendizado do modelo proposto justamente pela especificidade do objeto de estudo e do conjunto de dados utilizado. As métricas obtidas durante todos os processos de treino incremental e avaliação do modelo foram tabuladas e são apresentadas mais adiante no Capítulo 4.

3.3 PAINEL DE MONITORAMENTO GRAFANA

Para a produção do painel de monitoramento DNS partimos da coleta de consultas DNS no servidor recursivo da Reitoria Unesp, coletando e armazenando em banco de dados consultas DNS provenientes de todos os campus da universidade.

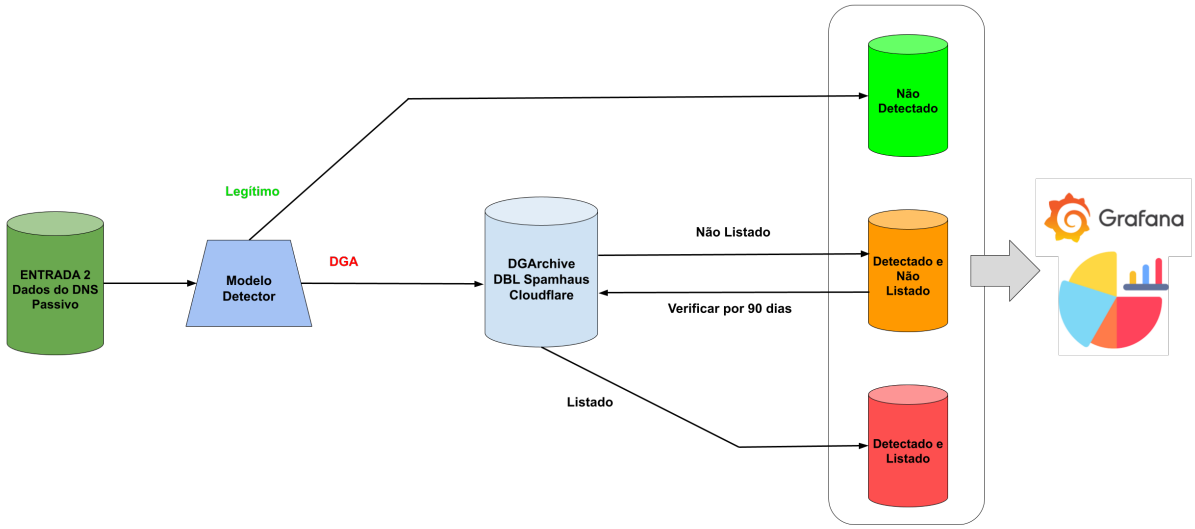
Para a geração dos Risk Score utilizamos o modelo de aprendizado profundo descrito anteriormente, o modelo detector foi treinado com os 1 milhão de exemplos da lista Majestic Million para exemplos de domínios legítimos, para exemplos DGA foi utilizada uma base de 15 dias de feed obtida por meio de API DGArchive, o modelo também foi treinado incrementalmente a cada 15 dias com um conjunto de novos exemplos DGA coletados do feed DGArchive.

Um script Python 3.8 foi desenvolvido para tratar diariamente todas as consultas DNS coletadas do dia anterior, separando exclusivamente o nome de domínio de segundo nível, transpor os caracteres para seu código ASCII e apresentar para o modelo detector treinado.

Domínios classificados como legítimos são alimentados em uma base de não-detectados, domínios classificados como DGA são então encaminhados para a consulta em listas de reputação a fim de validar sua condição maliciosa.

Cada domínio classificado como DGA é consultado em três listas, DGArchive, Cloudflare e DBL Spamhaus, caso este esteja presente em alguma das listas o domínio então é armazenado em uma base de detectados-listados, os domínios classificados como DGA, mas que não foram encontrados em nenhuma das listas são armazenados em uma base de detectados-não-listados, e são verificados nas listas novamente no dia seguinte e por um período de 90 dias. A Figura Figura 13 apresenta o processo completo de geração das bases de domínios e apresentação no painel de monitoramento.

Figura 13 – Processo de verificação de domínios a partir de coleta de DNS passivo.



Fonte: Elaborado pelo autor.

O painel de monitoramento, portanto, parte das três bases geradas e atualizadas diariamente, não-detectados, detectados-não-listados e detectados-listados. Essas informações são apresentadas no painel Grafana, juntamente com os valores de Risk Score, data de detecção e listagem em alguma lista negra, qual lista este foi encontrado e mais.

O mesmo painel também possui conexão direta com o banco de dados do DNS Passivo, portanto cruzamentos podem ser feitos como, por exemplo, encontrar outros domínios resolvidos para o mesmo endereço IP de um domínio apontado como DGA.

3.4 CONSIDERAÇÕES PARCIAIS

O presente trabalho aborda o problema de detecção de DGAs apresentando um modelo de aprendizado profundo baseado em CNN e PLN. Apresentamos o processo de treinamento e avaliação do modelo proposto com uma base de dados fixa, para viabilizar a comparação direta dos resultados com modelos recentes apresentados na literatura especializada.

Abordamos também o processo de aprendizado incremental e a técnica utilizada para evadir o esquecimento catastrófico, utilizando um conjunto de dados que possua exemplos novos mesclados a exemplos já apresentados ao modelo.

Ademais, apresentamos o processo para utilização do modelo desenvolvido em conjunto com coletas de consultas DNS e apresentação dos resultados de forma amigável em um painel desenvolvido com a ferramenta Grafana.

Observando o modelo e os processos até aqui apresentados percebemos que foi possível percorrer um caminho completo partindo da identificação de uma necessidade e oportunidade de pesquisa, o desenvolvimento de um modelo de inteligência artificial baseado em princípios e técnicas de vanguarda nessa área de pesquisa, a validação desse modelo e a aplicação deste em

ambiente de mundo real, servindo como ferramenta de cibersegurança em uma instituição de grande porte como a Unesp.

4 RESULTADOS

Este Capítulo apresenta os resultados obtidos com o modelo detector de DGA durante os experimentos realizados para o desenvolvimento dos mesmos. Os resultados do modelo proposto foram comparados com os resultados obtidos por uma variação do modelo completo e também com os resultados de modelos propostos na literatura recente sobre o mesmo conjunto de dados, a fim de aferirmos a eficácia das técnicas aplicadas ao modelo proposto e também em função da técnica de aprendizado incremental aplicada, com o objetivo de validar a metodologia aplicada.

4.1 AVALIAÇÃO DO MODELO PROPOSTO

Nesta seção são apresentados os resultados obtidos durante os testes do modelo proposto, cujo objetivo é classificar um domínio entre legítimo e DGA. A avaliação do modelo proposto foi feita em duas etapas, a primeira verificando a efetividade do próprio modelo em relação à aplicação combinada de rede convolucional e embedding, a segunda verificando sua desempenho com relação ao incremento de exemplos no decorrer do tempo.

4.1.1 Avaliação Inicial do Modelo Proposto

Para avaliar a eficiência do modelo proposto e da aplicação das técnicas de PLN aos nomes de domínio, principalmente a eficácia do uso de embedding em nível de caracteres, outro modelo derivado do modelo principal, que aqui chamaremos de Modelo 1, foi testado sobre o mesmo conjunto de dados, o modelo foi modificado removendo-se a camada embedding na entrada de dados, chamaremos este de Modelo 2.

Também foram treinados e testados com o mesmo conjunto de dados outros cinco modelos apresentados por trabalhos recentes, o modelo Bilbo apresentado em (Highnam et al., 2020), um modelo utilizando o transformer pré-treinado DistilBERT, uma versão leve e mais rápida do transformer BERT proposto no trabalho (Sanh et al., 2019), além de modelos simples CNN, CNN+LSTM e RNN propostos em (Ravi et al., 2023).

As estruturas dos modelos testados para comparação foram obtidas nos repositórios github de cada autor, citados como material anexo aos trabalhos. A Tabela 5 apresenta a estrutura básica de cada um dos modelos comparados sobre a mesma base de dados de treinamento e teste.

Tabela 5 – Modelos testados e suas estruturas básicas.

Modelo	Arquitetura
Modelo 1 (Proposto)	CNN+Embedding+ReLU+MaxPooling
Modelo 2 (Proposto)	CNN+ReLU+MaxPooling
Bilbo (Highnam et al., 2020)	Embedding+CNN+LSTM
DistilBERT (Sanh et al., 2019)	DistilBERT+DNN
CNN (Ravi et al., 2023)	CNN Simples
CNN+LSTM (Ravi et al., 2023)	CNN+LSTM Simples
RNN (Ravi et al., 2023)	RNN Simples

Fonte: Elaborada pelo autor.

Para cada modelo, o conjunto de dados contendo 15 dias de coletas de exemplos DGA por meio da API DGArchive e 1 milhão de exemplos legítimos obtidos do portal Majestic Million foi separado em 70% para treinamento e 30% para teste, treinado por 10 épocas utilizando a função *callback* para salvar a melhor acurácia na validação, o conjunto de validação foi selecionado proporcionalmente na ordem de 10% do conjunto de treinamento utilizando-se o parâmetro validation_split e as métricas geradas a partir da predição do conjunto de teste.

Esse processo foi repetido por três vezes, utilizando-se sorteios distintos para os conjuntos de treinamento e teste, seguindo o conceito de *repeated holdout*. As Tabelas 6, 7 e 8 apresentam a comparação das métricas obtidas pelos dois modelos propostos e pelos modelos de referência sobre os mesmos conjuntos de treinamento, validação e teste para cada um dos sorteios de seleção dos conjuntos de dados. A Tabela 9 apresenta a média aritmética das métricas obtidas por cada modelo nos três testes efetuados.

Tabela 6 – Resultados obtidos pelos modelos sobre o primeiro conjunto de treinamento e teste.

Modelo	Acurácia	Precisão	Recall	TFP
Modelo 1 (Proposto)	98,00%	97,95%	97,96%	2,31%
Modelo 2 (Proposto)	91,55%	91,50%	91,17%	11,38%
Bilbo (Highnam et al., 2020)	95,51%	95,33%	95,52%	4,42%
DistilBERT (Sanh et al., 2019)	94,65%	94,46%	94,61%	5,67%
CNN (Ravi et al., 2023)	96,14%	95,93%	96,22%	3,25%
CNN+LSTM (Ravi et al., 2023)	96,90%	96,76%	96,92%	2,94%
RNN (Ravi et al., 2023)	95,77%	95,53%	95,88%	3,39%

Fonte: Elaborada pelo autor.

Tabela 7 – Resultados obtidos pelos modelos sobre o segundo conjunto de treinamento e teste.

Modelo	Acurácia	Precisão	Recall	TFP
Modelo 1 (Proposto)	98,00%	97,96%	97,96%	2,35%
Modelo 2 (Proposto)	91,70%	91,51%	91,50%	9,81%
Bilbo (Highnam et al., 2020)	95,65%	95,47%	95,68%	4,19%
DistilBERT (Sanh et al., 2019)	94,49%	94,42%	94,28%	7,07%
CNN (Ravi et al., 2023)	96,08%	96,06%	95,91%	5,24%
CNN+LSTM (Ravi et al., 2023)	96,92%	96,87%	96,83%	3,81%
RNN (Ravi et al., 2023)	96,10%	96,00%	96,03%	4,48%

Fonte: Elaborada pelo autor.

Tabela 8 – Resultados obtidos pelos modelos sobre o terceiro conjunto de treinamento e teste.

Modelo	Acurácia	Precisão	Recall	TFP
Modelo 1 (Proposto)	98,00%	97,97%	97,94%	2,50%
Modelo 2 (Proposto)	91,68%	91,61%	91,35%	10,92%
Bilbo (Highnam et al., 2020)	95,73%	95,65%	95,60%	5,21%
DistilBERT (Sanh et al., 2019)	94,50%	94,44%	94,31%	6,96%
CNN (Ravi et al., 2023)	96,23%	96,17%	96,11%	4,67%
CNN+LSTM (Ravi et al., 2023)	96,91%	95,79%	96,88%	3,26%
RNN (Ravi et al., 2023)	95,92%	95,71%	95,98%	3,62%

Fonte: Elaborada pelo autor.

Tabela 9 – Resultados médios obtidos pelos modelos sobre os três conjuntos de treinamento e teste.

Médias				
Modelo	Acurácia	Precisão	Recall	TFP
Modelo 1 (Proposto)	98,00%	97,96%	97,95%	2,39%
Modelo 2 (Proposto)	91,64%	91,54%	91,34%	10,71%
Bilbo (Highnam et al., 2020)	95,63%	95,48%	95,60%	4,61%
DistilBERT (Sanh et al., 2019)	94,58%	94,44%	94,40%	6,57%
CNN (Ravi et al., 2023)	96,15%	96,05%	96,08%	4,39%
CNN+LSTM (Ravi et al., 2023)	96,91%	96,47%	96,88%	3,34%
RNN (Ravi et al., 2023)	95,93%	95,75%	95,96%	3,83%

Fonte: Elaborada pelo autor.

As métricas obtidas em cada um dos sorteios distintos se mostraram bem próximas, o que demonstra que o sorteio dos dados não exerceu impacto significativo nos resultados, sendo a média aritmética representativa para demonstrar a efetividade do modelo proposto.

Se compararmos as métricas obtidas pelo Modelo 1 proposto com as métricas obtidas pelo Modelo 2 proposto podemos perceber a efetividade da aplicação da técnica de PLN e embedding na entrada de dados do modelo, uma vez que o Modelo 1 obteve métricas cerca de 6% superiores ao Modelo 2, além de uma TFP próximo de cinco vezes menor.

Com relação aos demais modelos obtidos de trabalhos recentes, nosso Modelo 1 superou todos em todas as métricas sobre o mesmo conjunto de dados, com destaque para a TFP, que representa o percentual de domínios legítimos erroneamente classificados como DGA.

Em ambientes de mundo real, onde não há o balanceamento de classes, a quantidade de domínios legítimos é absolutamente muito maior que a de DGAs, portanto uma TFP alta pode tornar o número absoluto de falsos positivos maior que os verdadeiros positivos.

Esta comparação dos resultados diretos sobre o mesmo conjunto de dados entre modelos presentes na literatura atual e o Modelo 1 proposto nos permite conferir a robustez do Modelo 1 perante a literatura e diante da aplicação real sobre um conjunto de dados diversificado.

4.1.2 Avaliação do Processo de Treinamento Incremental

Como descrito na Seção 4.1.1, o Modelo 1 se mostrou equilibrado e com métricas competitivas comparadas a diversos outros modelos presentes na literatura. Em um segundo momento o mesmo foi testado em um processo de treinamento incremental, para isto foram coletadas 12 amostras de 15 dias de feed DGA utilizando a API DGArchive, portanto uma janela de 180 dias, compreendidos entre 22 de julho de 2023 e 18 de janeiro de 2024.

A cada etapa o modelo foi treinado a partir de seu estado atual com os exemplos de 15 dias concatenados com 500.000 exemplos legítimos provenientes da lista Majestic Million, usando a função *callback* para guardar a melhor acurácia na validação. O modelo treinado então foi aplicado aos exemplos coletados em todos os períodos de 15 dias subsequentes, concatenados aos 500.000 exemplos legítimos restantes e as métricas anotadas a fim de verificarmos a evolução do modelo. As Tabelas 10 e 11 apresentam respectivamente a evolução da acurácia e precisão para todos os conjuntos em relação a cada etapa de treinamento.

Tabela 10 – Evolução da acurácia do modelo em função de cada uma das etapas de treinamento.

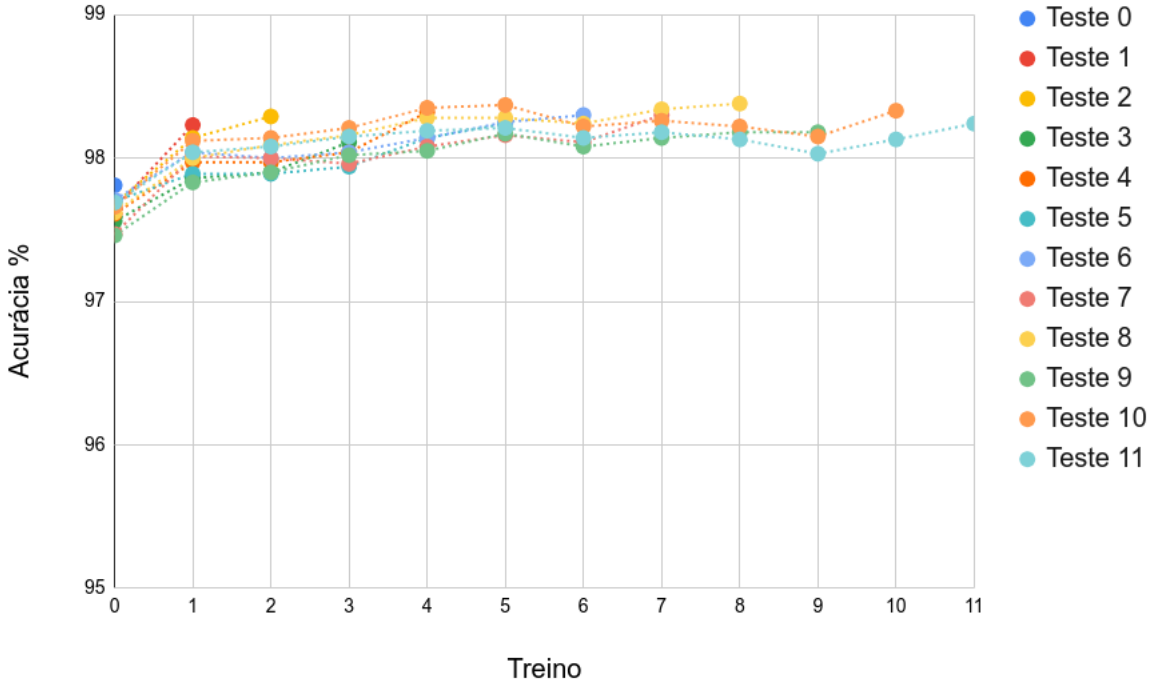
Treino	0	1	2	3	4	5	6	7	8	9	10	11
0	97,81%											
1	97,65%	98,23%										
2	97,68%	98,14%	98,29%									
3	97,56%	97,86%	97,90%	98,11%								
4	97,61%	97,97%	97,97%	98,04%	98,32%							
5	97,70%	97,89%	97,89%	97,94%	98,14%	98,24%						
6	97,71%	98,04%	98,00%	98,04%	98,14%	98,25%	98,30%					
7	97,47%	98,01%	98,00%	97,96%	98,08%	98,16%	98,11%	98,30%				
8	97,62%	98,00%	98,09%	98,16%	98,28%	98,28%	98,24%	98,34%	98,38%			
9	97,46%	97,83%	97,90%	98,02%	98,05%	98,17%	98,08%	98,14%	98,18%	98,18%		
10	97,66%	98,12%	98,14%	98,21%	98,35%	98,37%	98,22%	98,26%	98,22%	98,15%	98,33%	
11	97,69%	98,04%	98,08%	98,15%	98,19%	98,21%	98,14%	98,18%	98,13%	98,03%	98,13%	98,24%

Fonte: Elaborada pelo autor.

Uma variação positiva média de 0,17% a cada treinamento incremental foi percebida na acurácia do modelo durante o experimento, com destaque para o resultado obtido para a amostra temporalmente mais distante, que quando apresentada ao modelo após a primeira etapa de treinamento obteve uma acurácia de 97,69% e após todas as etapas de treinamento obteve acurácia de 98,24%, um ganho de 0,55%. A Figura Figura 14 apresenta graficamente a evolução

da acurácia do modelo com o decorrer de cada etapa de treinamento incremental. Visualmente podemos perceber a ascendência, mesmo que tênue, da métrica avaliada.

Figura 14 – Evolução da acurácia do modelo a cada etapa de treinamento incremental.



Fonte: Elaborado pelo autor.

Para podermos demonstrar o equilíbrio do modelo, além da acurácia, avaliamos a evolução da precisão do modelo a cada etapa de treinamento incremental. Da mesma forma podemos perceber para a precisão do modelo uma evolução positiva constante média de 0,41% a cada etapa de treinamento incremental.

Se olharmos exclusivamente a amostra temporalmente mais distante, esta obteve 96,53% de precisão quando apresentada ao modelo treinado com o primeiro conjunto de dados, e 98,20% quando apresentado ao modelo treinado incrementalmente em onze etapas, um ganho de 1,47% em relação à precisão.

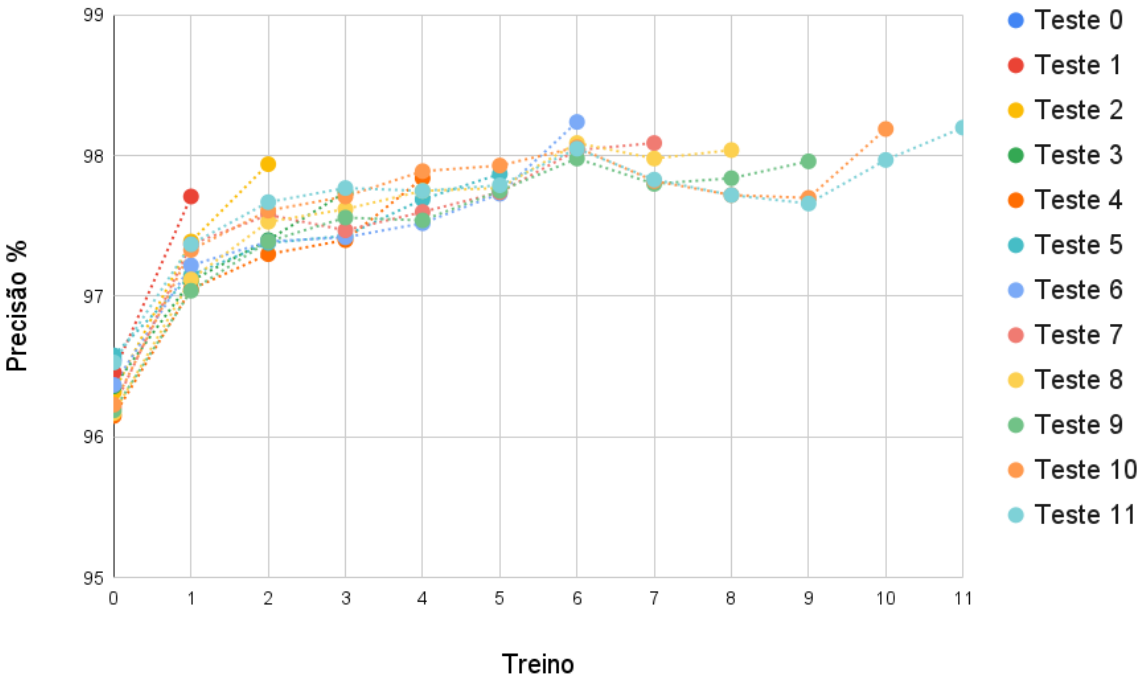
Tabela 11 – Evolução da precisão do modelo em função de cada uma das etapas de treinamento.

Treino	0	1	2	3	4	5	6	7	8	9	10	11
0	96,55%											
1	96,46%	97,71%										
2	96,32%	97,39%	97,94%									
3	96,36%	97,11%	97,40%	97,74%								
4	96,15%	97,05%	97,30%	97,40%	97,84%							
5	96,58%	97,15%	97,38%	97,43	97,69%	97,87%						
6	96,37%	97,22%	97,39%	97,42%	97,52%	97,73%	98,24%					
7	96,21%	97,37%	97,58%	97,47%	97,60%	97,74%	98,04%	98,09%				
8	96,17%	97,12%	97,53%	97,62%	97,75%	97,77%	98,09%	97,98%	98,04%			
9	96,19%	97,04%	97,39%	97,56%	97,54%	97,75%	97,98%	97,80%	97,84%	97,96%		
10	96,23%	97,33%	97,61%	97,71%	97,89%	97,93%	98,06%	97,82%	97,72%	97,70%	98,19%	
11	96,53%	97,37%	97,67%	97,77%	97,75%	97,79%	98,05%	97,83%	97,72%	97,66%	97,97%	98,20%

Fonte: Elaborada pelo autor.

A Figura Figura 15 apresenta graficamente a evolução da precisão do modelo no decorrer de cada etapa de treinamento incremental, novamente para esta métrica percebemos a curva ascendente do gráfico. Desta forma, com acurácia e precisão mantendo seus níveis ou mesmo evoluindo positivamente, mesmo que levemente, a cada novo treinamento incremental do modelo, podemos afirmar que a técnica de treinamento incremental do modelo se mostrou efetiva no sentido de evitar o esquecimento catastrófico do modelo.

Figura 15 – Evolução da precisão do modelo a cada etapa de treinamento incremental.



Fonte: Elaborado pelo autor.

Em nosso experimento, como já mencionado no início deste capítulo, a classe DGA recebeu recorrentemente novos exemplos a cada etapa de treinamento incremental, porém a classe legítima não recebeu novos exemplos por questões de restrições no conjunto de dados.

Sendo assim, calculamos a Taxa de Falsos Negativos (TFN) para cada etapa de treinamento incremental, a fim de verificarmos a evolução do modelo quanto à classe para a qual o modelo recebeu novos exemplos de maneira recorrente. A Tabela 12 apresenta as TFN para cada conjunto de DGA a cada etapa de treinamento incremental.

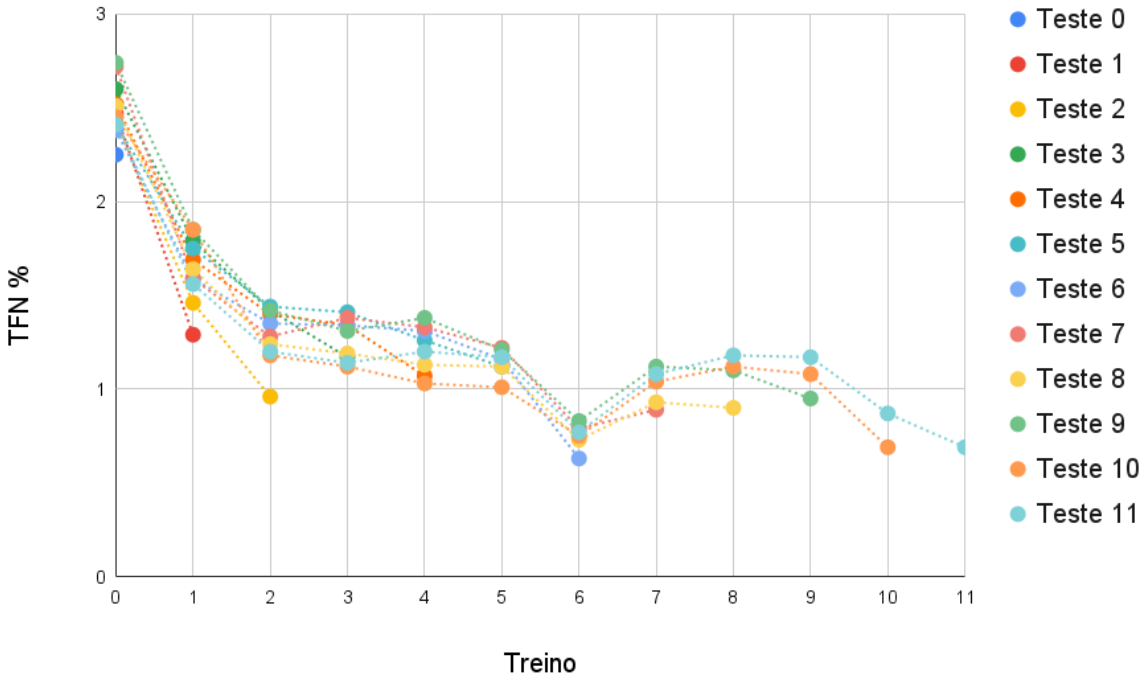
Tabela 12 – Evolução da TFN do modelo a cada etapa de treinamento incremental.

Treino	0	1	2	3	4	5	6	7	8	9	10	11
0	2,25%											
1	2,47%	1,29%										
2	2,42%	1,46%	0,96%									
3	2,6%	1,79%	1,42%	1,17%								
4	2,52%	1,69%	1,4%	1,34%	1,07%							
5	2,41%	1,75%	1,44%	1,41%	1,26%	1,12%						
6	2,38%	1,59%	1,35%	1,34%	1,31%	1,16%	0,63%					
7	2,72%	1,59%	1,28%	1,38%	1,33%	1,22%	0,79%	0,89%				
8	2,51%	1,64%	1,24%	1,19%	1,13%	1,12%	0,73%	0,93%	0,9%			
9	2,74%	1,85%	1,42%	1,31%	1,38%	1,21%	0,83%	1,12%	1,1%	0,95%		
10	2,45%	1,85%	1,18%	1,12%	1,03%	1,01%	0,75%	1,04%	1,12%	1,08%	0,69%	
11	2,41%	1,56%	1,2%	1,14%	1,2%	1,17%	0,77%	1,08%	1,18%	1,17%	0,87%	0,69%

Fonte: Elaborada pelo autor.

Novamente, se observarmos a amostra temporalmente mais distante, vemos que a TFN, portanto os exemplos DGA erroneamente classificados como legítimos, caiu de 2,41% na primeira etapa para 0,69% na última. A Figura Figura 16 apresenta graficamente essa evolução da TFN durante cada etapa do treinamento incremental.

Figura 16 – Evolução da TFN do modelo a cada etapa de treinamento incremental.

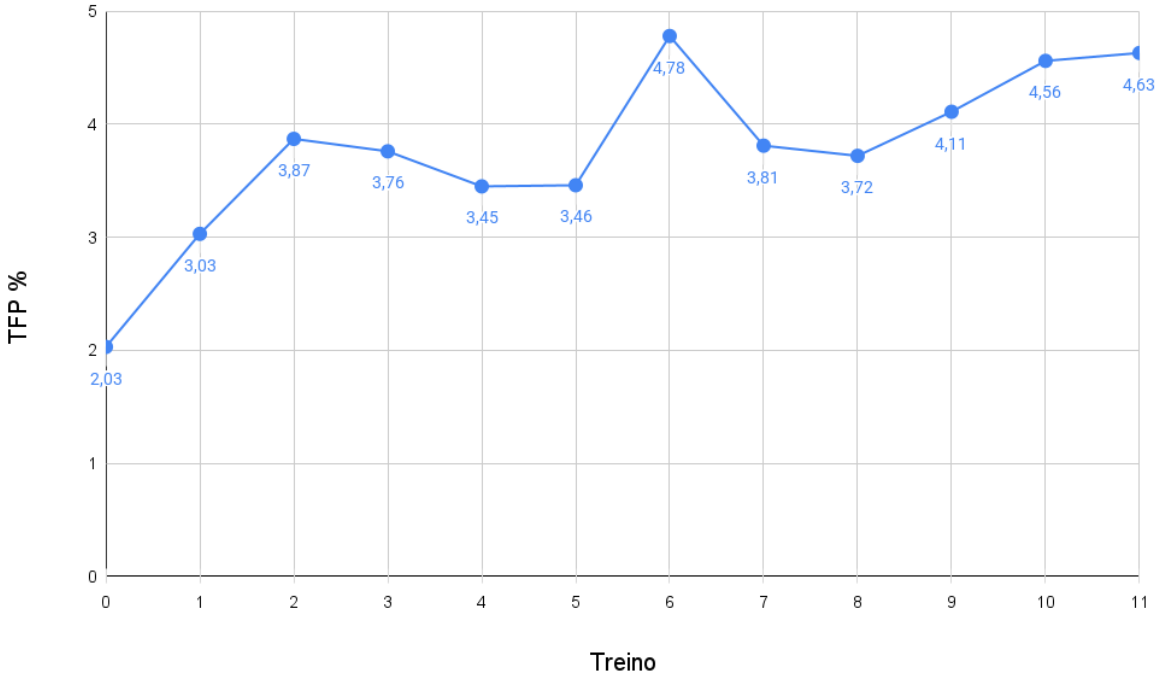


Fonte: Elaborado pelo autor.

Percebe-se uma evolução negativa da TFN, o que significa que o modelo passou a identificar mais exemplos DGA de maneira correta, essa característica é positiva no sentido de que menos exemplos DGA passaram desapercebidos pelo modelo à medida que este foi treinado incrementalmente com novos exemplos.

Da mesma forma, decidimos calcular a TFP a cada nova etapa de treinamento incremental, a fim de aferir o comportamento do modelo para a classe para a qual não houve o fornecimento de novos exemplos. A Figura Figura 17 apresenta graficamente a evolução da TFP a cada etapa de treinamento.

Figura 17 – Evolução da TFP do modelo a cada etapa de treinamento incremental.



Fonte: Elaborado pelo autor.

Nota-se uma evolução da TFP, saltando de 2,03%, no primeiro ciclo de treinamento, para 4,63% no último ciclo de treinamento. Dessa forma, a TFP piorou consideravelmente com os treinamentos incrementais, apesar de se manter dentro dos limites encontrados na literatura recente.

Sendo assim, o processo de treinamento incremental parece conseguir manter a eficiência média do modelo, evitando quedas bruscas das métricas gerais, que seriam indícios de esquecimento catastrófico.

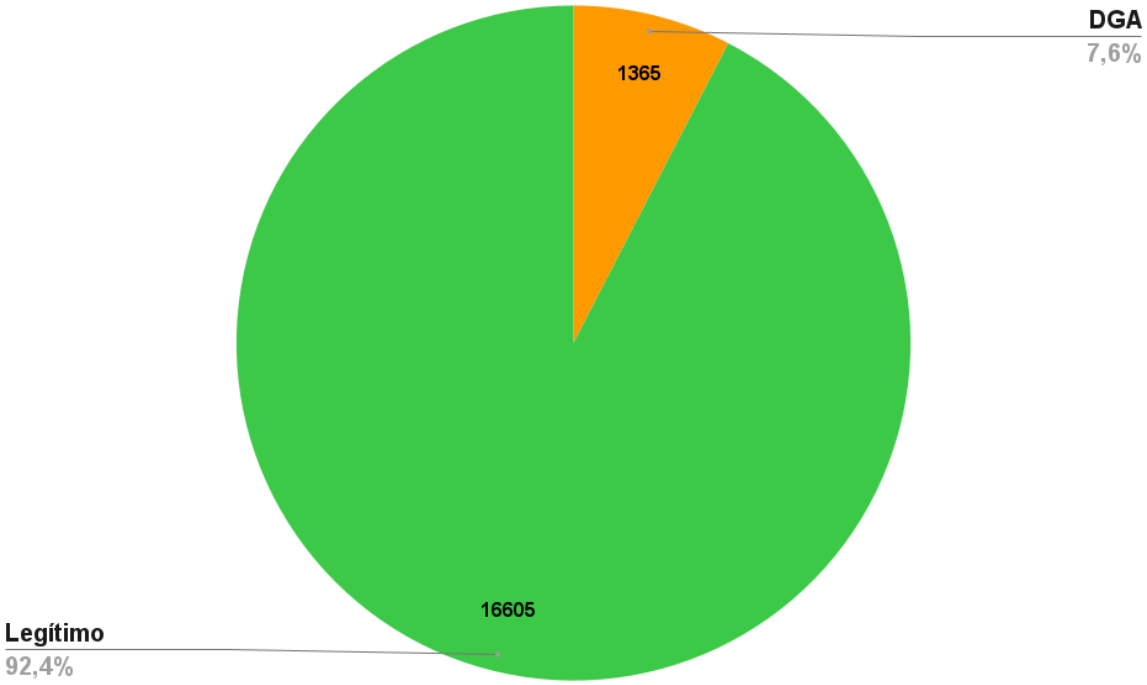
A apresentação recorrente de novos exemplos DGA ao modelo pareceu impactar positivamente a detecção dessa classe, com a TFN caindo de maneira constante durante as etapas do experimento. Por outro lado, a falta de novos exemplos para uma das classes, legítimos, acabou por impactar em sua detecção de forma correta, causando um aumento da TFP, o que certamente configura um desafio a ser transposto.

4.2 APLICAÇÃO EM DNS PASSIVO E PAINEL DE MONITORAMENTO

Para a avaliação de performance do modelo proposto em ambiente de mundo real selecionamos as consultas únicas capturadas por meio de DNS passivo que já tinham encerrado o período de 90 dias de verificações nas listas de bloqueio utilizadas pelo painel, na data de 05 de maio de 2024, portanto todos os domínios visualizados antes de 05 de fevereiro de 2024 atendiam a esta condição.

Um total de 17.970 domínios de segundo nível únicos foram apresentados ao modelo neste período, destes, 1.365 foram apontados como sendo DGA pelo modelo e 16.605 foram apontados como legítimos. Após o período de quarentena de 90 dias, 141 domínios foram confirmados em alguma lista de verificação. A Figura Figura 18 apresenta um gráfico com a proporção entre detectados e não detectados no período analisado.

Figura 18 – Proporção entre domínios detectados e não detectados após 90 dias de conferência em listas de bloqueio.



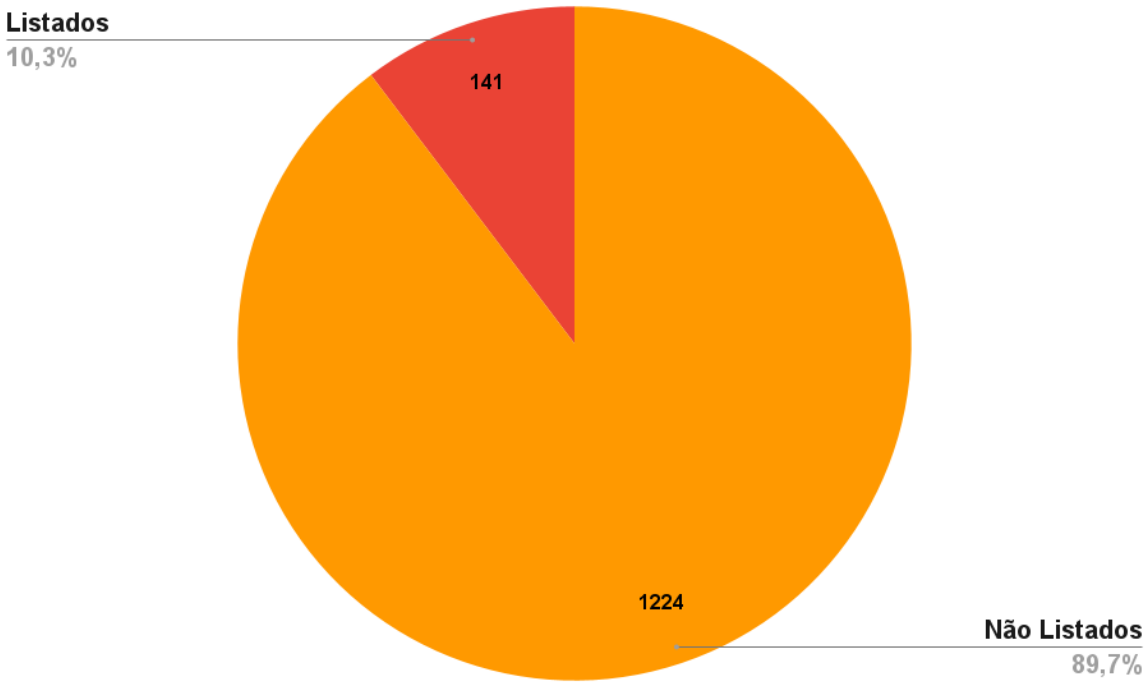
Fonte: Elaborado pelo autor.

Como podemos perceber, no mundo real o desbalanceamento de classes DGA e legítimos é bastante importante, sendo o conjunto de DGAs muito menor que o conjunto de legítimos, com 7,6% dos domínios únicos apontados como DGA pelo modelo proposto e 92,4% apontados como legítimos.

Os domínios apontados como DGA pelo modelo proposto foram submetidos à conferência em listas de bloqueio confiáveis durante um período de 90 dias conforme citado anteriormente.

A Figura Figura 19 apresenta a proporção entre domínios detectados que foram encontrados listados em alguma das listas utilizadas e os não listados.

Figura 19 – Domínios detectados listados e não listados em listas de bloqueio após 90 dias.



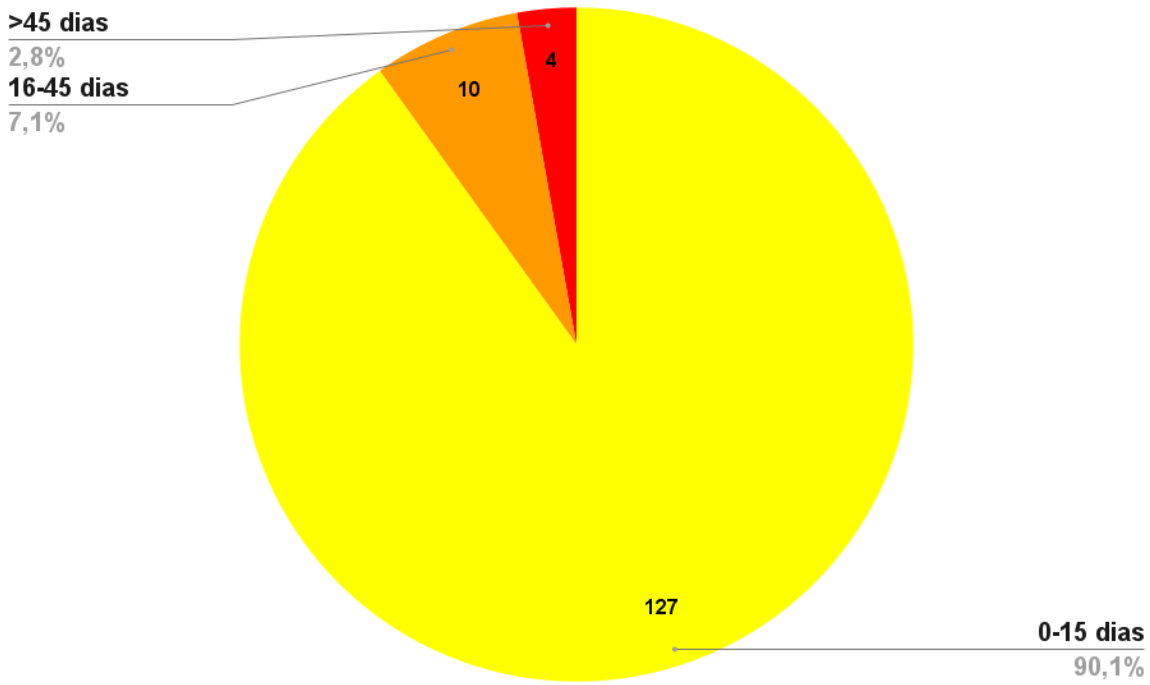
Fonte: Elaborado pelo autor.

Para o período analisado, 141 domínios, ou 10,30%, dos domínios apontados como DGA foram encontrados em alguma lista de bloqueio durante o período de 90 dias após este ser coletado pelo DNS passivo, enquanto 1224 domínios, ou 89,70%, não foram encontrados listados nas listas de bloqueio utilizadas. Significa dizer que a maior parte dos DGA percebidos pelo modelo proposto não estavam em nenhuma das listas de bloqueio utilizadas, mostrando a efetividade da aplicação da técnica apresentada neste trabalho em detectar DGAs, mesmo antes destes serem reportados como maliciosos.

Para domínios apontados pelo modelo e encontrados em alguma das listas de bloqueio, foram contabilizados quantos dias após o domínio ser apontado pelo modelo como sendo DGA este viria a ser encontrado em uma lista de bloqueio.

A Figura Figura 20 apresenta a distribuição entre menos de 15 dias, de 16 a 45 dias e mais de 45 dias em quarentena. Percebemos que 127, ou 90,10%, dos DGA encontrados listados em alguma lista de bloqueio foram encontrados listados entre 0 e 15 dias após serem identificados, 10, ou 7,10%, foram encontrados listados entre 16 e 45 dias, e outros 4, ou 2,80%, foram encontrados listados mais de 45 dias após serem detectados.

Figura 20 – Distribuição de domínios listados por tempo em quarentena.



Fonte: Elaborado pelo autor.

Esta análise nos permite verificar que o modelo proposto conseguiu identificar diretamente no tráfego de rede coletado, ocorrências de DGA com certa antecedência se comparado às listas de bloqueio utilizadas. Esta antecipação representa uma vantagem importante para a ferramenta de cibersegurança proposta em relação a métodos tradicionais baseados em listas de bloqueio.

Como vimos na Seção 2.2, DGAs são utilizados em todos os momentos do ciclo de vida de um ataque que utiliza botnets. Sendo assim, essa detecção antecipada em relação às listas públicas, é particularmente importante para auxiliar na mitigação de um ataque antes mesmo que este ocorra, com a identificação de estações já exploradas e que estejam mantendo comunicação com o C2 do atacante, a fim de evoluir no processo de exploração.

4.3 FALSOS POSITIVOS

Esta seção tem por objetivo aproximar a análise e discutir os falsos positivos apontados pelo modelo proposto quando aplicado em ambiente de mundo real e suas implicações, uma vez que essa métrica foi a que obteve os piores resultados durante o experimento com treinamentos incrementais.

4.3.1 Analisando Falsos Positivos

Um falso positivo ocorre quando um domínio legítimo é incorretamente classificado como DGA pelo modelo. Em outras palavras, o modelo identifica características compatíveis com

DGA em um nome de domínio legítimo. Apesar de comum em sistemas que utilizam análise automatizada de padrões, o falso positivo pode causar problemas quando não tratado ou abordado corretamente, gerando alarmes falsos, bloqueios inadequados e até mesmo indisponibilidades indesejadas.

Como apresentado anteriormente, durante os experimentos de aplicação do modelo proposto em ambiente de mundo real, foram analisados 17.970 nomes de domínio únicos, dos quais 1.365 foram apontados como DGA pelo modelo e 16.605 apontados como legítimos. Se utilizarmos a TFP média de 2,39%, obtida nos experimentos de produção do modelo proposto, sobre estes números, podemos estimar o número absoluto de FP, TP e a Precisão do modelo quando aplicado em ambiente de mundo real.

Uma vez que, para obtermos o número absoluto de FP precisamos do número absoluto de TN, e este segundo é calculado também com base no número absoluto de TP; e que somente uma verificação manual de cada nome de domínio observado poderia aferir com certeza sua condição de DGA ou legítimo, consideramos que o número absoluto de TN é igual ao número de domínios apontados como legítimos pelo modelo, mesmo sabendo que o número correto seja um pouco menor, e que isso impacta negativamente na Precisão obtida.

Sendo assim, do grupo de 1.365 domínios apontados como DGA pelo modelo, cerca de 407 seriam domínios legítimos, portanto FP, os demais 958 seriam realmente DGA, o que resulta em uma Precisão estimada de 70,18% para o modelo quando aplicado em dados de tráfego DNS real.

Essa diferença de precisão do modelo quando utilizado em bases de dados de exemplo e quando aplicado em mundo real está diretamente relacionada ao desbalanceamento de classes presentes no mundo real, onde menos de 8% dos domínios capturados foram identificados como DGA, portanto mesmo uma TFP baixa impacta significativamente nos resultados.

Ainda, decidimos observar mais de perto alguns exemplos de domínios classificados erroneamente como DGA pelo modelo e, para estes, uma verificação manual foi feita, a fim de validar sua condição de domínio legítimo.

A Tabela 13 apresenta alguns exemplos de domínios legítimos que foram erroneamente classificados como DGA, seu Risk Score, se este foi encontrado listado em alguma lista pública, se existe um website online e a data de registro do domínio.

Tabela 13 – Exemplos de domínios legítimos erroneamente classificados como DGA.

Domínio	Risk Score	Listado	Website Online	Registrado em
jcsveiculos.com.br	0,52	Não Listado	Sim	16/08/2016
dtm.com.br	0,66	DGArchive	Não	26/06/2001
dscadmbenseimoveis.com.br	0,79	Não Listado	Sim	04/10/2023
fmup.com.br	0,80	Não Listado	Sim	11/06/2021
jhsf.com.br	0,99	Spamhaus DBL	Sim	09/02/2000

Fonte: Elaborada pelo autor.

O dado Risk Score, como visto anteriormente, diz respeito ao valor atribuído pelo Modelo 1

a um determinado nome de domínio, que varia entre 0 e 1, sendo maior a probabilidade de um domínio ser DGA quanto mais próximo de 1 o seu Risk Score for.

Durante o experimento, todos os domínios identificados como DGA pelo modelo foram verificados em listas públicas de domínios maliciosos; também buscamos esse dado para análise.

Primeiramente, observamos o próprio nome de domínio, que é o único dado utilizado pelo modelo proposto para determinar se este é um domínio legítimo ou DGA. Podemos verificar que há domínios de comprimentos distintos, alguns possuem palavras conhecidas do idioma português, mas outros possuem apenas uma sigla. Uma característica comum a todos é a presença de sequências de mais de duas consoantes.

Com relação ao Risk Score, verifica-se que há domínios legítimos que receberam valores mais baixos e outros com valores bem próximos de 1. Como quanto maior o Risk Score, mais características de DGA o modelo encontrou em um determinado nome de domínio, podemos concluir que, em determinados falsos positivos, o modelo errou por menos, ou seja, atribuiu um Risk Score menor ao domínio, e em outros errou por mais, atribuindo um Risk Score muito alto para um domínio legítimo.

A verificação em listas públicas foi utilizada neste trabalho para validar parte das detecções feitas pelo modelo em ambiente de mundo real. Porém, apesar de muitos dos falsos positivos realmente não terem sido encontrados em nenhuma das listas, há casos em que um domínio legítimo foi encontrado em listas públicas de domínios maliciosos.

Ainda, para os domínios que não foram encontrados em listas públicas de bloqueio de domínio, efetuamos uma verificação na ferramenta de compatibilidade de expressões regulares DGA, oferecida pelo DGArchive. Todos os três domínios não listados apresentaram compatibilidade de expressão regular com ao menos duas famílias de DGA.

Também verificamos manualmente a existência de um website publicado e a data de registro do nome de domínio. Observa-se que apenas um nome de domínio não possuía um website publicado. Sendo o domínio mais antigo registrado em 09/02/2000 e o mais recente registrado em 04/10/2023.

Percebemos, portanto, que somente uma análise manual foi conclusiva para estes casos onde a classificação automatizada falhou. Para isso, foi necessária a utilização de mais dados e detalhes do domínio verificado.

4.4 CONSIDERAÇÕES PARCIAIS

Com base nos resultados obtidos e nas análises efetuadas sobre os mesmos, podemos considerar que o modelo proposto neste trabalho apresentou resultados importantes durante os experimentos laboratoriais, com conjuntos de dados fixos, com resultados alinhados e até mesmo superiores a modelos publicados na literatura recente.

Os resultados obtidos durante os experimentos com treinamentos incrementais também foram satisfatórios, uma vez que métricas relevantes não sofreram impactos negativos, até mesmo

evoluindo em algum nível. Podemos considerar que o modelo conseguiu adaptar-se aos novos exemplos DGA sem perder sua capacidade de identificar exemplos anteriormente utilizados nos treinamentos.

Esse ponto é particularmente importante quando pensamos na utilização do modelo em mundo real por um longo período, uma vez que novas ameaças surgem diariamente e os algoritmos geradores de domínio se atualizam e evoluem a fim de evadirem as camadas e mecanismos de segurança.

Porém, a falta de novos exemplos da classe legítima acabou por impactar negativamente a TFP, que aumentou consecutivamente, mesmo mantendo-se em níveis alinhados com os trabalhos recentemente publicados.

Podemos considerar também que a TFP talvez seja uma das métricas mais importantes quando abordamos a detecção automatizada de DGA em tráfego DNS de mundo real, uma vez que há um desbalanceamento grande entre as classes, DGA e legítimo, nesse tipo de conjunto de dados.

Uma análise mais cuidadosa e individualizada de falsos positivos pôde demonstrar que o modelo se engana em casos diversos; domínios legítimos de variados comprimentos e morfologias foram identificados, erroneamente, como DGA. Encontramos até mesmo domínios legítimos encontrados nas listas públicas de domínios maliciosos, o que reforça a dificuldade metodológica do tema abordado neste trabalho.

5 CONCLUSÃO

Este capítulo apresenta as conclusões obtidas durante o processo de desenvolvimento deste trabalho, as sugestões para trabalhos futuros e a evolução da abordagem proposta, além de apresentar as produções técnicas e científicas obtidas durante o desenvolvimento deste trabalho.

5.1 CONCLUSÕES

Os DGA fazem parte da infraestrutura de ataque de ameaças cibernéticas importantes e estão presentes em ataques de grande destaque e impacto financeiro, sendo um desafio a sua contenção em função do volume de nomes de domínio gerados pelas ameaças e por se utilizarem de forma espúria de um serviço basilar da internet que é o DNS.

Ainda, por se tratar de uma ferramenta fundamental para a operação de diversas ameaças, incluindo botnets, os DGAs evoluem constantemente com o surgimento de novas famílias ou modificando seu código gerador de nomes de domínios, portanto sua identificação automatizada deve acompanhar essa dinâmica para evitar sua obsolescência precoce.

Este trabalho se focou no desenvolvimento de um modelo de aprendizado profundo, com técnicas de PLN para detectar automaticamente os DGA, utilizando como entrada somente os nomes de domínio como texto curto, sem o desenvolvimento manual de características. Também, visando a manutenção e melhoria das métricas do modelo proposto, foi aplicada uma técnica de aprendizado incremental ao modelo, com exemplos coletados durante 180 dias do feed DGArchive.

Um painel de monitoramento DNS, utilizando DNS passivo, foi implementado utilizando a ferramenta Grafana para a produção de gráficos e apresentação dos resultados obtidos a partir da aplicação do modelo proposto em coletas de consultas DNS de mundo real. Embora este painel de monitoramento configure uma aplicação prática para o modelo apresentado neste trabalho, as possibilidades não se limitam a essa aplicação. Um modelo com métricas adequadas pode ser integrado em firewalls para ativamente filtrar saídas para endereços suspeitos.

O mesmo pode ser também utilizado por registradores, para verificar o nome de domínio antes deste ser publicado, já que o modelo apresentado neste trabalho utiliza como entrada apenas o nome, este pode ser analisado no momento de registro, não sendo necessária sua publicação e acompanhamento para que se suspeite de sua finalidade.

Podemos ainda pensar na integração do modelo em ferramentas de proteção de *endpoints*, analisando as requisições do próprio dispositivo por domínios DGA, o que pode caracterizar o dispositivo como um botclient.

A aplicação das técnicas de PLN e o uso de embedding em nível de caractere se mostrou eficiente para melhorar as métricas em cerca de 7%, ficando o modelo proposto completo com acurácia média de 98,00% e precisão de 97,96%. Os resultados obtidos pelo modelo proposto

foram superiores em todas as métricas quando comparados com diversos modelos publicados por trabalhos recentes sobre o mesmo conjunto de dados, o que demonstra a robustez do modelo proposto e seu alinhamento com o estado da arte das pesquisas que buscam detectar DGA com técnicas de aprendizado profundo.

O experimento com aprendizado incremental por transferência de aprendizado demonstrou que o modelo foi capaz de melhorar suas métricas à medida que recebeu novos exemplos de DGA somados a exemplos já conhecidos, a fim de evitar o esquecimento catastrófico. A acurácia do modelo evoluiu positivamente em média 0,17% a cada etapa de treinamento incremental e a precisão evoluiu positivamente em média 0,41% a cada treinamento incremental.

A técnica se mostrou eficiente, portanto, em evitar o fenômeno do esquecimento catastrófico e manter os níveis de detecção elevados. Apesar disso, a falta de fornecimento de novos exemplos legítimos ao modelo durante o processo incremental acabou por impactar negativamente a TFP, que evoluiu de 2,03% para 4,64%, o que pode configurar um problema quando este for aplicado em dados de mundo real.

A aplicação do modelo em dados coletados em ambiente de mundo real demonstrou que as técnicas propostas neste trabalho colaboraram grandemente na detecção dos DGAs, principalmente por dois fatores: primeiro, por detectarem, em sua maioria, exemplos nunca adicionados em listas de bloqueio de reputação e, portanto, não detectáveis por técnicas que se utilizam exclusivamente de listas externas de verificação. Segundo, por detectar DGAs com certa antecedência, mesmo em relação aos que em algum momento foram encontrados em listas de bloqueio, o que também caracteriza uma vantagem estratégica do ponto de vista da cibersegurança.

Verificamos também que a precisão do modelo em ambiente de mundo real tende a diminuir em função do desbalanceamento de ocorrências de domínios legítimos e DGA, sendo este segundo um volume muito menor que o do primeiro. Portanto, mesmo um modelo com uma TFP baixa, em ambientes reais, tende a apresentar uma quantidade relativamente grande de falsos positivos em números absolutos, e esse volume será tão maior quanto a diferença de consultas legítimas e DGA do ambiente analisado. Em nosso ambiente, a precisão estimada em mundo real com uma TFP de 2,39% obtida nos testes do modelo foi de 70,18%.

O painel de monitoramento Grafana produzido a partir da aplicação do modelo proposto neste trabalho em coletas de consultas DNS no ambiente da Universidade Estadual Paulista se mostrou uma ferramenta útil para a equipe de cibersegurança e segurança de redes de computadores da universidade, e está hoje sendo utilizado em produção pela equipe, já com certas melhorias em relação ao apresentado neste trabalho.

Dessa forma, concluímos que o trabalho obteve resultados relevantes para a aplicação de técnicas de aprendizado profundo, PLN e aprendizado incremental, além de proporcionar a aplicação em mundo real do modelo desenvolvido, integrando-o a uma ferramenta de DNS passivo e apresentando os resultados de forma acessível e rica em informações por meio de um painel Grafana.

5.2 LIMITAÇÕES E TRABALHOS FUTUROS

Durante a avaliação dos resultados, principalmente durante a abordagem detalhada de falsos positivos, podemos perceber que, no conjunto de dados de mundo real, os falsos positivos tendem a ser numericamente volumosos, mesmo com a TFP de apenas 2,39%. Esse aspecto inviabiliza, por exemplo, a utilização do modelo para bloqueios automatizados de firewall.

A verificação manual de alguns falsos positivos nos deram pistas sobre características léxicas presentes em alguns nomes de domínio que favoreceram sua classificação equivocada por parte do modelo, como a presença de sequências de consoantes.

Ainda, alguns exemplos selecionados constavam em listas públicas de domínios maliciosos, mesmo possuindo um site publicado e sendo claramente um domínio legítimo. Outros possuíam compatibilidade de expressão regular com mais de uma família DGA. Tal ocorrência demonstra a dificuldade metodológica da abordagem automatizada de detecção de DGA, uma vez que mesmo os grandes feeds possuem falsos positivos não tratados.

A confirmação da legitimidade dos exemplos selecionados somente foi possível com uma verificação manual, com o cruzamento de dados whois e o acesso ao site publicado naquele domínio. Sendo assim, as abordagens de detecção automatizadas, mesmo as que se utilizam de tecnologias de vanguarda como as técnicas propostas neste trabalho, ainda não estão prontas para serem utilizadas sem a devida supervisão humana.

Para trabalhos futuros podemos sugerir estudos no sentido de reduzir as TFP em modelos detectores a fim de melhorar seu desempenho quando aplicados no mundo real onde as classes costumam ser muito desbalanceadas.

Também, o modelo proposto neste trabalho aplica uma classificação binária, DGA e legítimo, portanto o desenvolvimento de modelos para classificação multiclasse, identificando a qual família de ameaça pertence cada DGA detectado pode ser de grande valor para equipes de cibersegurança e segurança de redes que terão melhor visão sobre qual o tipo de ameaça estão confrontando.

Podemos sugerir abordagens com técnicas de aprendizado de máquina não supervisionado a fim de atacar a identificação automatizada de novas famílias DGA quando estas surgirem e ainda não estiverem catalogadas, o que elevaria o nível da detecção e classificação dos DGA de maneira significativa.

Possivelmente, a utilização de dados complementares ao próprio nome de domínio possa ser uma alternativa para ratificar a condição maliciosa desses domínios, como dados whois e/ou dados referentes a consultas DNS do domínio analisado.

Adicionalmente, podemos sugerir abordagens explicáveis de inteligência artificial, a fim de compreender melhor as razões ou características que levam cada modelo a apontar determinado domínio como DGA. Isso pode ajudar na identificação de pontos importantes e nortear o desenvolvimento de modelos mais precisos.

Este trabalho apresentou um painel de monitoramento de tráfego DNS que apresenta os

resultados obtidos pelo modelo desenvolvido somado a algum enriquecimento, como a verificação de um domínio detectado em listas públicas de domínios maliciosos. Porém, as utilizações de modelos de detecção automatizada podem e devem ir além.

Integrações e outros cruzamentos de dados podem ser feitos, a fim de disponibilizar para análise de especialista, endereços IP de possíveis C2, além de avaliar a adição automatizada ou semi-supervisionada destes em regras de firewall, por exemplo.

REFERÊNCIAS

ABADI, M. et al. **TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems**. 2015. Software available from tensorflow.org. Disponível em: https://www.tensorflow.org/.

ANIL, N. K.; KAUR, B. P. Analyzing sentiments with machine learning and deep learning through word embedding methods. In: **2024 IEEE 3rd World Conference on Applied Intelligence and Computing (AIC)**. [S.l.: s.n.], 2024. p. 705–710.

ANTONAKAKIS, M. et al. Understanding the mirai botnet understanding the mirai botnet. **26th USENIX Security Symposium**, 2022. Disponível em: https://www.researchgate.net/publication/363475096_Understanding_the_Mirai_Botnet_Understanding_the_Mirai_Botnet.

AZHARI, R. G. et al. The detection of mirai botnet attack on the internet of things (iot) device using support vector machine (svm) model. In: **2022 10th International Conference on Information and Communication Technology (ICoICT)**. [S.l.: s.n.], 2022. p. 397–401.

Bambenek Consulting OSINT. **DGA Feed**. [S.l.]: Bambenek Consulting OSINT, 2023. https://www.bambenekconsulting.com/.

BAO, Z.; WANG, W.; LAN, Y. Using passive dns to detect malicious domain name. In: **Proceedings of the 3rd International Conference on Vision, Image and Signal Processing**. New York, NY, USA: Association for Computing Machinery, 2020. (ICVISP 2019). ISBN 9781450376259. Disponível em: https://doi.org/10.1145/3387168.3387236.

C, C. et al. Improved domain generation algorithm to detect cyber-attack with deep learning techniques. In: **2022 IEEE 2nd Mysore Sub Section International Conference (MysuruCon)**. [s.n.], 2022. p. 1–8. Disponível em: https://ieeexplore.ieee.org/abstract/document/9972526.

CHOWDHARY, K. R. Natural language processing. In: ____. **Fundamentals of Artificial Intelligence**. New Delhi: Springer India, 2020. p. 603–649. ISBN 978-81-322-3972-7. Disponível em: https://doi.org/10.1007/978-81-322-3972-7_19.

Ciso Advisor. **Brasil já ocupa o 2º lugar no ranking mundial de ransomware**. [S.l.]: Ciso Advisor, 2023. https://www.cisoadvisor.com.br/brasil-ja-ocupa-o-2o-lugar-no-ranking-mundial-de-ransomware/.

CLARK, J. H. et al. Pre-training an efficient tokenization-free encoder for language representation. **Transactions of the Association for Computational Linguistics**, MIT Press, v. 10, p. 73–91, 2022. Disponível em: https://doi.org/10.1162%2Ftacl_a_00448.

DALLI, A. Impact of hyperparameters on deep learning model for customer churn prediction in telecommunication sector. **Mathematical Problems in Engineering**, Hindawi, v. 2022, p. 4720539, Feb 2022. ISSN 1024-123X. Disponível em: https://doi.org/10.1155/2022/4720539.

Data Science Academy. **Deep Learning Book**. [S.l.]: https://www.deeplearningbook.com.br/, 2023. https://www.deeplearningbook.com.br/.

DEVLIN, J. et al. BERT: pre-training of deep bidirectional transformers for language understanding. **CoRR**, abs/1810.04805, 2018. Disponível em: http://arxiv.org/abs/1810.04805.

Fraunhofer FKIE. **DGArchive**. [S.l.]: Fraunhofer Institute for Communication, Information Processing and Ergonomics, 2023. https://dgarchive.caad.fkie.fraunhofer.de/.

GERSHENSON, C. Artificial neural networks for beginners. **CoRR**, cs.NE/0308031, 2003. Disponível em: http://arxiv.org/abs/cs/0308031.

GOGOI, B.; AHMED, T. Dga domain detection using pretrained character based transformer models. In: **2023 IEEE Guwahati Subsection Conference (GCON)**. [s.n.], 2023. p. 01–06. Disponível em: https://ieeexplore.ieee.org/abstract/document/10183602.

GOODFELLOW, I.; BENGIO, Y.; COURVILLE, A. **Deep Learning**. [S.l.]: MIT Press, 2016. http://www.deeplearningbook.org.

Grafana Labs. **Grafana**. [S.l.]: Grafana, 2023. https://grafana.com/.

HIGHNAM, K. et al. Real-time detection of dictionary DGA network traffic using deep learning. **CoRR**, abs/2003.12805, 2020. Disponível em: https://arxiv.org/abs/2003.12805.

HUANG, W. et al. Pepc: A deep parallel convolutional neural network model with pre-trained embeddings for dga detection. In: **2022 International Joint Conference on Neural Networks (IJCNN)**. [s.n.], 2022. p. 1–8. Disponível em: https://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=9892081.

KAMBOURAKIS, G. et al. **Botnets: Architectures, Countermeasures, and Challenges**. 1. ed. Boca Raton: CRC Press, 2019. ISBN 9780429329913.

KAMIL, S. et al. The rise of ransomware: A review of attacks, detection techniques, and future challenges. In: **2022 International Conference on Business Analytics for Technology and Security (ICBATS)**. [S.l.: s.n.], 2022. p. 1–7.

Kepios Pte. Ltd. **DataReportal: 2025**. [S.l.]: DataReportal, 2025. https://datareportal.com/reports/digital-2025-global-overview-report.

KINGMA, D. P.; BA, J. **Adam: A Method for Stochastic Optimization**. 2017.

KOUTSOUKAS, A. et al. Deep-learning: investigating deep neural networks hyper-parameters and comparison of performance to shallow methods for modeling bioactivity data. **Journal of Cheminformatics**, v. 9, n. 1, p. 42, Jun 2017. ISSN 1758-2946. Disponível em: https://doi.org/10.1186/s13321-017-0226-y.

KRUTI, A.; BUTT, U.; SULAIMAN, R. B. **A review of SolarWinds attack on Orion platform using persistent threat agents and techniques for gaining unauthorized access**. 2023.

LLOPIS-IBOR, L. et al. Fast incremental learning by transfer learning and hierarchical sequencing. **Expert Systems with Applications**, v. 212, p. 118580, 2023. ISSN 0957-4174. Disponível em: https://www.sciencedirect.com/science/article/pii/S0957417422016402.

Maarten Wullink, Moritz Muller, Marco Davids, Giovane C. M. Moura, and Cristian Hesselman. ENTRADA: Enabling DNS Big Data Applications. In: **APWG Symposium on Electronic Crime Research (eCRIME 2016), Toronto, ON, Canada. June 1, 2, and 3, 2016**. [S.l.: s.n.], 2016.

Majestic. **Majestic Million**. [S.l.]: Majestic, 2023. https://pt.majestic.com/reports/majestic-million.

MANIRIHO, P.; MAHMOOD, A. N.; CHOWDHURY, M. J. M. Api-maldetect: Automated malware detection framework for windows based on api calls and deep learning techniques. **Journal of Network and Computer Applications**, v. 218, p. 103704, 2023. ISSN 1084-8045. Disponível em: https://www.sciencedirect.com/science/article/pii/S1084804523001236.

MCCULLOCH, W. S.; PITTS, W. (1943) Warren S. McCulloch and Walter Pitts, A logical calculus of the ideas immanent in nervous activity, Bulletin of Mathematical Biophysics 5: 115-133. In: **Neurocomputing, Volume 1: Foundations of Research**. The MIT Press, 1988. ISBN 9780262267137. Disponível em: https://doi.org/10.7551/mitpress/4943.003.0004.

MUKHAMEDIEV, R. I. et al. From classical machine learning to deep neural networks: A simplified scientometric review. **Applied Sciences**, v. 11, n. 12, 2021. ISSN 2076-3417. Disponível em: https://www.mdpi.com/2076-3417/11/12/5541.

MUSSIRALIYEVA, S. et al. Creating a model of semantic analysis of extremist texts in the kazakh language. **Journal of Mathematics, Mechanics and Computer Science**, v. 121, n. 1, p. 110–121, Apr. 2024. Disponível em: https://bm.kaznu.kz/index.php/kaznu/article/view/1332.

NAGARIKAR, A. et al. Incremental learning of classification models in deep learning. In: **Proceedings of the 6th International Conference on Advances in Artificial Intelligence**. New York, NY, USA: Association for Computing Machinery, 2023. (ICAAI '22), p. 56–60. ISBN 9781450396943. Disponível em: https://doi.org/10.1145/3571560.3571568.

NAJEH, H.; LOHR, C.; LEDUC, B. Convolutional neural network bootstrapped by dynamic segmentation and stigmergy-based encoding for real-time human activity recognition in smart homes. **Sensors**, v. 23, n. 4, 2023. ISSN 1424-8220. Disponível em: https://www.mdpi.com/1424-8220/23/4/1969.

NAYEF SITI NORUL HUDA SHEIKH ABDULLAH, R. S. A. M. S. B. H. Text extraction with optimal bi-lstm. **Computers, Materials & Continua**, v. 76, n. 3, p. 3549–3567, 2023. ISSN 1546-2226. Disponível em: http://www.techscience.com/cmc/v76n3/54348.

NetLab 360. **NetLab360**. [S.l.]: Network Security Research Lab at 360, 2022. https://data.netlab.360.com/.

Núcleo de Informação e Coordenação do Ponto BR. **IX.br: Estatísticas**. [S.l.]: IX.br, 2025. https://ix.br/agregado/.

Núcleo de Informação e Coordenação do Ponto BR. **Registro.br: Estatísticas**. [S.l.]: Registro.br, 2025. https://registro.br/dominio/estatisticas/.

PARK, K. H. et al. Unsupervised malicious domain detection with less labeling effort. **Computers Security**, v. 116, p. 102662, 2022. ISSN 0167-4048. Disponível em: https://www.sciencedirect.com/science/article/pii/S016740482200061X.

RAMESH, R.; CHAUDHARI, P. **Model Zoo: A Growing "Brain"That Learns Continually**. 2022.

RAVI, V. et al. Adversarial defense: Dga-based botnets and dns homographs detection through integrated deep learning. **IEEE Transactions on Engineering Management**, v. 70, n. 1, p. 249–266, 2023.

RUMELHART, D. E.; MCCLELLAND, J. L.; GROUP, P. R. **Parallel Distributed Processing, Volume 1: Explorations in the Microstructure of Cognition: Foundations**. The MIT Press, 1986. ISBN 9780262291408. Disponível em: https://doi.org/10.7551/mitpress/5236.001.0001.

SALEHIN, I.; KANG, D.-K. A review on dropout regularization approaches for deep neural networks within the scholarly domain. **Electronics**, v. 12, n. 14, 2023. ISSN 2079-9292. Disponível em: https://www.mdpi.com/2079-9292/12/14/3106.

SANH, V. et al. Distilbert, a distilled version of BERT: smaller, faster, cheaper and lighter. **CoRR**, abs/1910.01108, 2019. Disponível em: http://arxiv.org/abs/1910.01108.

SARTORELLO, L. **DNS: o que é, como funciona e qual escolher?** [S.l.]: Alura, 2023. https://www.alura.com.br/artigos/dns-o-que-e-qual-escolher.

SCHILLER, C. A. et al. **Botnets: The Killer Web App**. [S.l.]: Syngress Publishing, 2011. ISBN 9780080500232.

SegInfo. **Mirai Botnet – Visão geral de suas Ameaças e Mitigações**. [S.l.]: SegInfo – Portal, Podcast e Evento sobre Segurança da Informação, 2021. https://seginfo.com.br/2021/08/23/mirai-botnet-visao-geral-de-suas-ameacas-e-mitigacoes/.

SHAHZAD, H.; SATTAR, A.; SKANDARANIYAM, J. Dga domain detection using deep learning. In: . [S.l.: s.n.], 2021. p. 139–143.

SILVEIRA, M. R. et al. Detection of newly registered malicious domains through passive dns. In: **2021 IEEE International Conference on Big Data (Big Data)**. [S.l.: s.n.], 2021. p. 3360–3369.

Singapore Computer Society. **Simplifying the difference: Machine Learning vs Deep Learning**. [S.l.]: Singapore Computer Society, 2020. https://www.scs.org.sg/articles/machine-learning-vs-deep-learning.

Statista. **Statista: Digital Trend**. [S.l.]: Statista, 2023. https://www.statista.com/studies-and-reports/digital-and-trends.

SUN, X.; LIU, Z. Domain generation algorithms detection with feature extraction and domain center construction. **PLOS ONE**, Public Library of Science, v. 18, p. 1–25, 01 2023. Disponível em: https://doi.org/10.1371/journal.pone.0279866.

TANENBAUM, A. S.; FEAMSTER, N.; WETHERALL, D. **Redes de Computadores**. 6. ed. Porto Alegre: Pearson Education do Brasil, 2021. ISBN 9780135408001.

TRISNA, K. W. et al. From context-independent embedding to transformer: Exploring sentiment classification in online reviews with deep learning approaches. **J. Theor. Appl. Inf. Technol**, v. 102, p. 6980–7003, 2024.

TUAN, T. A.; LONG, H. V.; TANIAR, D. On detecting and classifying dga botnets and their families. **Computers Security**, v. 113, p. 102549, 2022. ISSN 0167-4048. Disponível em: https://www.sciencedirect.com/science/article/pii/S0167404821003734.

TZU, S. **The Art of War**. Mineola, NY: Dover Publications, 2002. Tradução inglesa de Lionel Giles (tradução original 1910). Trecho citado traduzido ao português. Obra original: c. 5th century BCE.

VEN, G. M. van de; TUYTELAARS, T.; TOLIAS, A. S. Three types of incremental learning. **Nature Machine Intelligence**, v. 4, n. 12, p. 1185–1197, Dec 2022. ISSN 2522-5839. Disponível em: https://doi.org/10.1038/s42256-022-00568-3.

WEI, X. et al. Automatic multi-label diagnosis of single-lead ecg using novel hybrid residual recurrent convolutional neural networks. **Biomedical Signal Processing and Control**, v. 95, p. 106422, 2024. ISSN 1746-8094. Disponível em: https://www.sciencedirect.com/science/article/pii/S1746809424004804.

WONG, A. D. **Detecting Domain-Generation Algorithm (DGA) Based Fully-Qualified Domain Names (FQDNs) with Shannon Entropy**. 2023.

YANG, L. et al. Detecting word-based algorithmically generated domains using semantic analysis. **Symmetry**, v. 11, n. 2, 2019. ISSN 2073-8994. Disponível em: https://www.mdpi.com/2073-8994/11/2/176.

YANG, Q.; GU, Y.; WU, D. Survey of incremental learning. In: **2019 Chinese Control And Decision Conference (CCDC)**. [S.l.: s.n.], 2019. p. 399–404.

ZHANG, A. et al. **Dive into Deep Learning**. [S.l.]: Cambridge University Press, 2023. https://D2L.ai.

DADOS CURRICULARES

IDENTIFICAÇÃO	
	João Rafael Gregório 25/05/1984
Nacionalidade	brasileiro
Nome em citações bibliográficas	Gregório, João Rafael Gregório, João
Currículo Lattes	http://lattes.cnpq.br/3662040070806398
ORCID	https://orcid.org/0000-0001-7783-2567
Outro identificador	URL
FORMAÇÃO ACADÊMICA	
2010/2011	Especialização em Redes de Computadores e Comunicação de Dados. Universidade Estadual de Londrina - UEL
2002/2006	Licenciatura em Análise de Sistemas e Tecnologias. Faculdade de Tecnologia de Ourinhos - FATEC
PRODUÇÃO BIBLIOGRÁFICA	
GREGÓRIO, JOÃO RAFAEL; CANSIAN, ADRIANO MAURO ; NEVES, LEANDRO ALVES . Class Incremental Deep Learning: A Computational Scheme to Avoid Catastrophic Forgetting in Domain Generation Algorithm Multiclass Classification. Applied Sciences-Basel, v. 14, p. 7244, 2024. GREGÓRIO, JOÃO; CANSIAN, ADRIANO ; NEVES, LEANDRO ; SALVADEO, DENIS . Deep Convolutional Neural Network and Character Level Embedding for DGA Detection. In: 26th International Conference on Enterprise Information Systems, 2024, Angers. Proceedings of the 26th International Conference on Enterprise Information Systems, 2024. p. 167.	
PARTICIPAÇÃO EM BANCAS E ORIENTAÇÕES	
Bancas de trabalhos de conclusão	
Orientações	
PARTICIPAÇÃO EM EVENTOS CIENTÍFICOS	