Universidade de Brasília
Instituto de Ciências Exatas
Departamento de Ciência da Computação

Modelos de inteligência artificial para detecção de atividades humanas focados na detecção de Advanced Persistent Threat (AI-APT)

Gabriel A. Castro
Leonardo R. Souza

Monografia apresentada como requisito parcial
para conclusão do Bacharelado em Ciência da Computação

Orientador
Prof. Dr. Li Weigang

Brasília
2022

Universidade de Brasília
Instituto de Ciências Exatas
Departamento de Ciência da Computação

Modelos de inteligência artificial para detecção de atividades humanas focados na detecção de Advanced Persistent Threat (AI-APT)

Gabriel A. Castro
Leonardo R. Souza

Monografia apresentada como requisito parcial
para conclusão do Bacharelado em Ciência da Computação

Prof. Dr. Li Weigang (Orientador)
CIC/UnB

Prof. Dr. Edison Ishikawa
Universidade de Brasília

Prof. Dr. Jorge Henrique Cabral Fernandes
Universidade de Brasília

Prof. Dr. Marcelo Grandi Mandelli
Coordenador do Bacharelado em Ciência da Computação

Brasília, 29 de setembro de 2022

Dedicatória

Gabriel Alves Castro - Este trabalho não poderia ter ocorrido sem toda a ajuda e motivação que sempre recebi dos meus pais Solange e Cleibe e minha família, estes sempre fizeram sacrifícios enormes em prol da oportunidade privilegiada que tive de estudar nos melhores centros de ensino, e a fim de possibilitar que eu pudesse dedicar horas a fio aos meus estudos e sonhos. Não existe nenhuma árvore, grande e bonita, sem que ela seja regada e incentivada, "cresça, e toque o sol". Certamente nesta família fui incentivado a tocar o sol em todas as sementes que plantei, e isso não tem preço.

Como somos todos aqueles com quem vivemos, eu não posso deixar de agradecer a todos os meus amigos que me acompanharam até aqui, e à minha namorada Nathália. Sempre fui muito incentivado por todos, e pude manter uma vida saudável e leve, o que permitiu muitos dos sucessos que vivi até agora, inclusive o sucesso de ser como eu sou, no âmbito profissional e pessoal.

Por último, eu dedico esse trabalho a todos os grandes mestres que tive ao longo da vida, todos os professores, parentes e amigos. Ser professor e pesquisador no Brasil, tem sido um grande ato de coragem e nós como alunos, devemos responder com a nossa excelência e gratidão. O futuro do país está logo ali, vamos pegá-lo.

Leonardo Rodrigues de Souza - Somos frutos de todos que nos ajudaram a chegar até aqui. Foi uma longa caminha, apesar de ser apenas o início, mas também, extremamente gratificante. Não poderia ter esse privilégio sem o apoio de meus pais, Ivete e Francisco. Também agradeço ao apoio de todos os meus amigos e colegas que me acompanharam nesses anos da universidade, e em todos os anos de escola. Agradeço ao apoio e incentivo da minha namorada, Thalia. E aos meus mestres e professores, que sempre me apoiaram e me incentivaram em todos os meus anos nesta universidade.

Agradecimentos

Gabriel Alves Castro - Foram inúmeras as pessoas que influenciaram este trabalho e a minha trajetória profissional. Eu sou muito grato por ter a família que tenho, e por sempre ter tido a oportunidade de compartilhar a minha vida com pessoas incríveis com quem tive a oportunidade de conviver. Estas pessoas foram sem dúvida uma grande influência para a minha formação e eu agradeço imensamente por isso. Ao mesmo tempo, eu tive grandes mentores ao longo da minha trajetória aos quais devo muito conhecimento e gratidão por sua capacidade enorme em influenciar a minha formação. Agradeço ao professor Jorge Henrique por me aceitar para um projeto de iniciação científica, mesmo eu estando tão jovem, e por ter me acompanhado por vários anos durante a minha formação, me ensinando conhecimentos que vou levar para o resto da minha vida. Agradeço ao professor Ricardo Sampaio por ter me acompanhado em uma fase mais madura no campo de ciência de dados, e ter me ajudado enormemente em minha evolução . Também não poderia deixar de agradecer-lo por todos as oportunidades que me deu, sempre acreditando no meu trabalho. Agradeço ao professor Flávio por ter me acompanhado durante a minha iniciação no campo da teoria da computação com tanta atenção e paciência, o que permitiu grandes avanços em meus projetos de pesquisa. Agradeço ao professor Li Weigang ao lado do coronel James Martins por terem me permitindo ter contato com pesquisas de alto nível e conhecer novos caminhos no campo de inteligência artificial, este trabalho me empurrou todos os dias em direção à minha carreira no campo de engenharia de inteligências artificiais, e sou muito grato por isso. Espero para o futuro, fazer papel relevante nesta área, e ajudar o nosso país, a ser um pouco melhor do que é agora.

Leonardo Rodrigues de Souza - O presente trabalho é fruto de muito apoio e ensinamentos. Por isso, preciso agradecer a todos que fizeram parte da minha jornada até aqui. Meus pais, pelo apoio e por me darem o privilégio do poder de estudar. Agradeço aos meus professores, por todo o seu tempo dedicados ao ensino, que procurei aproveitar da melhor forma possível. Agradeço a minha namorada, Thalia, por todo seu apoio até aqui. E por fim, agradeço imensamente ao meu professor, mestre e orientador Li Weigang por todo o tempo investido na orientação deste, e de outros projetos. Agradeço também ao coronel James Martins por toda a ajusta e mentoria prestada.

Resumo

Advanced Persistent Threat (APT), ou ameaça persistente avançada e Inteligência Artificial (IA) (inteligências artificiais) são dois assuntos que vêm ganhando cada vez mais a atenção da comunidade científica. Diversas organizações possuem os seus dados fragilizados devido à contextos de APT, dentre eles, o da exfiltração de dados. A partir da difusão de ambientes de nuvem, os problemas aumentaram, tornando cada vez mais difícil a detecção e contra-medidas contra casos de exfiltração de dados. Nesse contexto, as tecnologias de IA e Aprendizagem de Máquina (AM) podem se tornar ferramentas indispensáveis no combate à esse tipo de ameaça, mas, ao mesmo tempo, essas tecnologias são desafiadas pela complexidade exigida por um ataque de exfiltração de dados aos modelos. Há uma quantidade de dados enorme neste contexto, e exemplos escassos de ataque, o que o torna um exemplo de problema de classificação extremamente complexo para qualquer técnica. Muitos dados, alto desbalanceamento de classes e comportamentos sutis e não identificáveis claramente nos dados. É neste cerne que este trabalho se introduz, propomos a construção de um ambiente de desenvolvimento de soluções para problemas de APT, a partir do uso do Aprendizagem Profunda (AP) e Meta-Learning (MetL). Foi realizada uma extensa revisão da literatura, com foco em sistemas de DLPS, mapeando o estado da arte da aplicação de algoritmos de machine learning no contexto da exfiltração de dados. Em seguida, foram selecionados os modelos mais relevantes para a construção de algoritmos de detecção de exfiltração de dados: Redes densas com uso de entropia, redes diferenciais robustas e transformers. Uma tecnologia de Meta-Learning (MetL) foi escolhida e estudada para conferir qualidade à construção dos modelos, e permitir a criação futura de processos de *autoML*. Um banco de dados de envio de pacotes pela nuvem foi criado, visando construir um banco de dados de casos de exfiltação de dados. Foi selecionado um banco de dados no domínio RAH para validar os modelos modelados, uma vez que os casos de exfiltração de dados também se encontram neste domínio. Os modelos foram testados no banco de dados criado de pacotes, levando à conclusão de que este ainda era muito pequeno para ser utilizado. Os modelos foram então testados no banco de dados RAH, chegando-se à seleção do modelo transformer como o mais promissor para solucionar o problema, chegando a 90% de acurácia. As redes robustas ainda

não foram descartadas, sendo sugeridas algumas mudanças para trabalhos futuros. Os objetivos gerais e específicos do trabalho foram alcançados, permitindo a construção de uma base sólida para a pesquisa e desenvolvimento do sistema de DLPS em parceria com o exército brasileiro.

Palavras-chave: IIA, APT, Meta-learning, One-shot learning, Few-shot learning, AI-APT, OOD, Redes densas, Redes robustas

Abstract

Advanced Persistent Threat (APT) and Artificial Intelligences are subjects that concenrs each time more the scientific community. Several organizations have his data exposed from APT threats, like the problem of data exfiltration. By the diffusion of clouds environments, more problems arrive, making the detection and countermeasures to data exfiltration much more complexes. therefore, Artificial Intelligence and Machine learning technologies could be the solutions for several problems, but, at the same time, these technologies are challenged by the complexity of the subject. There are a huge amount of data, few examples of attacks in the context of data exfiltation, what make it a problem of classification very complex. Huge amount of data, high inequality between classes, and soft behaviors hard to identify in data. Therefore, in this work, we propose the construction of a new environment for the creation of solutions for APT problems, by the use of Deep Learning and Meta-Learning (MetL). We showed, and we validated the algorithms in know Data Banks and we built the path for the TRANSLAB team creating algorithms able to identify a data exfiltration attack new for the scientific community. It was made an extense literature review, with focus on DLPS, mapping the state of the art in the application of machine learning algorithms in the context of data exfiltration. Therefore, the most relevant models for the construction of exfiltration detection algorithms were selected: Dense Networks with use of entropy, Robust Differential Networks and Transformers. A MetL tool was selected and studied for improving the quality of the models construction, and supply the future construction of *AutoML* process. A Packages Send By The Cloud Data Bank was built with the aim of the construction of an Exfiltration Data Bank. It was selected a data bank in the HAR context to validate the modeled models, as the Exfiltration Cases are in that field. The models had been tested with the created data bank, showing that the data bank was too small yet to be used. The models were tested with the HAR data bank, demonstrating that the Transfomer model is the most promising option to solve the problem, classifying with 90% of precision. The Robust Network was not discarded yet, being suggested some changes for future works. the general and specific objectives were reached, what was enough to the construction of a solid basis for the research and development of the DLPS system by a partnership with

the Brazilian army.

Keywords: IIA, APT, Meta-learning, One-shot learning, Few-shot learning, AI-APT, OOD, Dense Neural Network

Sumário

1 Introdução **1**
1.1 Motivação e defesa cibernética 3
1.2 Objetivos 5
1.3 Organização do trabalho 6

2 Revisão da Literatura **7**
2.1 Aprendizagem de Máquina (AM) e Aprendizagem Profunda (AP) 7
2.2 Reconhecimento de atividade humana e dados fora da distribuição 10
2.3 Redes densas 11
2.4 O transformer 12
2.5 Redes robustas 14
2.5.1 Classificação com redes robustas de sinais temporais 16
2.6 Meta-Learning (MetL) 19
2.6.1 Algoritmo genético 20
2.6.2 AutoML 21
2.7 Few-Shot Learning (FSL) 22
2.8 Entropia 22
2.9 Métricas de validação 23

3 Técnicas de Treinamento, Desenvolvimento e Validação **25**
3.1 Meta-Learning (MetL) 25
3.1.1 Algoritmo genético 27
3.1.2 AutoML 28
3.2 Few-Shot Learning (FSL) 28
3.3 Entropia 29
3.4 Métricas de validação 30

4 Metodologia **31**
4.1 Pesquisa 31
4.2 Confecção 32

4.3 Bancos de dados de referência 34
4.4 Argumentos para a seleção dos modelos 35

5 Caracterização do Ataque e Defesa Cibernéticos 37
5.1 Contexto da ameaça persistente avançada 37
5.2 Relação com o dataset UCI RAH 39
5.3 Testes de envio de dados pelo terminal ou navegador 40

6 Implementação 41
6.1 Uso do Meta-Learning (MetL) 41
6.2 Arquitetura TCP/IP 43
6.3 RD genética com entropia, para séries temporais 44
6.4 Transformer adaptado do text-embbeding 46
6.5 Redes robustas para a classificação de seis classes 47

7 Resultados 49
7.1 Distribuição do dado 49
7.2 O transformer é promissor 51
7.3 As redes robustas falharam 52
7.4 As redes densas falharam 53
7.5 Discussão e comparações 54

8 Conclusão e Trabalhos Futuros 56
8.1 Contribuições 56
8.2 Trabalhos futuros 57

Referências 59

Lista de Figuras

1.1 *MFog*: Ciclo de Vida de Exfiltração de Dados (CVED) [1] 3

2.1 Revisão da literatura 8
2.2 Modelo de DNN de exemplo 11
2.3 Arquitetura do *transformer, Attention Is all You Need* [2] 14
2.4 Esquema de classificação com redes robustas [3] 18

4.1 Ciclo de vida da análise do tráfego de dados 34
4.2 Giróscopio em um Smartphone 35

6.1 Abordagem proposta para detecção de pacotes maliciosos 45
6.2 Pré-processamento para o modelo UCI 45

7.1 Histograma da distribuição de exemplos por classe no dataset, dados de treinamento 50
7.2 Histograma da distribuição de exemplos por classe no dataset, dados de validação 51

Lista de Tabelas

7.1 Distribuição das classes dos dados de treinamento. 50
7.2 Distribuição das classes dos dados de teste. 50
7.3 Matriz de confusão dos *transformers*. 52
7.4 Comparação entre os modelos. 55

Lista de Abreviaturas e Siglas

AD Aprendizagem Diferencial.

AE Algoritmo Evolutivo.

AG Algoritmo Genético.

AM Aprendizagem de Máquina.

AP Aprendizagem Profunda.

APT Advanced Persistent Threat.

AWS Amazon Web Services.

BiLSTM Bidirectional Long Short-Term Memory.

BOSL Bayesian One-Shot Learning.

CDC Centro de Defesa Cibernética.

CEDCC Centro de Excelência em Defesa Cibernética Cooperativa.

ComDC Comando de Defesa Cibernética.

CR Computação por Reservatório.

CRISPM Cross-industry standard process for data mining.

CVED Ciclo de Vida de Exfiltração de Dados.

DA Data Augmentation.

DFD Dados Fora de Distribuição.

DLPS Data Lacking Protection System.

DNN Dense Neural Network.

DNS Domain Name System.

EMFA Estado-Maior das Forças Armada.

EUA Estados Unidas da América.

FA Forças Armadas.

FEM Fórum Econômico Mundial.

FSL Few-Shot Learning.

GCN Graph Convolutional Networks.

HDFS Hadoop Distributed File System.

IA Inteligência Artificial.

IoT Internet of Things.

IP Internet Protocol.

ISA Interconexão de sistemas abertos.

MetL Meta-Learning.

OL Once Learning.

OSL One-Shot Learning.

OTAN Organização do Tratado do Atlântico Norte.

PND Política Nacional de Defesa.

RAH Reconhecimento de Atividade Humana.

RD Redes Diferenciais.

RDD Redes Diferenciais Densas.

RELU Rectified Linear Unit.

RMLP Redes de Memórias de Longo Prazo.

RN Redes Neurais.

RR Redes Recorrentes.

SBAAC Sistema de Busca Avançado de Ameaças Cibernéticas.

SMDC Sistema Militar de Defesa Cibernética.

SMTP Simple Mail Transfer Protocol.

TCP/IP Transmission Control Protocol/Internet Protocol Suite.

TransLab Laboratório de Modelo Computacional para Transporte Aéreo.

Capítulo 1

Introdução

A segurança e a defesa do espaço cibernético brasileiro têm sua importância definida na Política Nacional de Defesa (PND), sendo essenciais para a garantia do funcionamento dos sistemas de informações, de gerenciamento e de comunicações de interesse nacional. Neste sentido, as expressões do Poder Nacional são representadas pela Capacidade de Proteção dentre as outras Capacidades Nacionais de Defesa, i.e., Pronta-resposta, Dissuasão, Coordenação e Controle, Gestão da Informação, Logística, Mobilidade Estratégica, Mobilização e Desenvolvimento Tecnológico de Defesa.

A Capacidade de Proteção, na área de segurança e defesa do espaço cibernético, como marco legal, envolve o desenvolvimento do Sistema Militar de Defesa Cibernética (SMDC), suas normas afins, bem como seu preparo e emprego, em todos os níveis. O SMDC é um "conjunto de instalações, equipamentos, doutrina, procedimentos, tecnologias, serviços e pessoal essenciais para realizar as atividades de defesa no espaço cibernético, assegurando, de forma conjunta, o seu uso efetivo pelas Forças Armadas (FA), bem como impedindo ou dificultando sua utilização contra interesses da Defesa Nacional".

A concepção do SMDC possui quatro estruturas fundamentais: Estado-Maior das Forças Armada (EMFA), o Comando de Defesa Cibernética (ComDC), o Centro de Defesa Cibernética (CDC) e as estruturas de defesa cibernética das Forças Armadas (FA), representada pelos Centros de Tratamento de Incidentes de Redes. Operacionalmente, o CDCiber realiza as ações de coordenação e de integração do Setor Cibernético nas FA, privilegiando, sempre que possível, atuações conjuntas.

Nesse contexto, o Comando de Defesa Cibernética (ComDC) juntamente com o CDC, nos últimos anos, conceberam o Sistema de Busca Avançado de Ameaças Cibernéticas (SBAAC), que tem por objetivo identificar ameaças cibernéticas avançadas, i.e., Advanced Persistent Threat (APT). Atualmente, tanto no universo acadêmico, quanto no universo operacional, identificar este tipo de ameaça ainda é algo extremamente desafiador e necessário, tendo em vista o alto grau de sofisticação da ameaça e os danos à instituição

que podem vir a ocorrer [4] [5].

Destarte, pesquisas recentes revelaram que a aplicação de Inteligência Artificial (IA), especificamente Aprendizagem de Máquina (AM) e Aprendizagem Profunda (AP), vem oferecendo possibilidades de detectar e identificar APT [4, 6, 1].

O SMDC ainda não adquiriu a capacidade de detectar e identificar APT e dados os prejuízos que podem vir a ser causados ao Estado Brasileiro e suas Instituições por ataques de APT, projetos como este, que se propõem a oferecer um ambiente de pesquisa e investigação na aplicação de IA para identificar ameaças cibernéticas avançadas, apresentam-se como oportunos.

Atualmente em virtude da pandemia de coronavírus a principal forma de contato entre as pessoas vem sendo pelas mídias digitais. A vida das pessoas cada vez mais se mistura com a vida virtual, raramente é necessário ir até um banco ou uma loja física. Empresas como o facebook, whatsapp e instagram proporcionam possibilidades de relacionamento interpessoal. Neste contexto, essas empresas usam sistema de *cloud computing* para garantir a eficiência e eficácia dessas aplicações. Apesar de suas vantagens, *cloud computing* traz vários riscos de segurança para seus usuários [7]. Os autores de [1] relataram que ataques de exfiltração de dados afeta todos os tipos de agentes, indivíduos, empresas públicas e privadas, e até mesmo organizações governamentais, custando em média, até aquele ano, pelo menos US $3,86 milhões por incidente. Os autores completam que esse tipo de incidente pode afetar profundamente a reputação de até mesmo gigantes do mercado. Podendo, até mesmo, revelar segredos de segurança nacional.

Em [8] os autores definem a exfiltração de dados como o processo de recuperação, cópia e transferência de dados sem a devida autorização. [1] expõe que esse tipo de violação de dados tem aumentado substancialmente nos últimos anos, de 779 em 2015 para 1632 em 2017. Mesmo gigantes do mercado estão vulneráveis a esse tipo de ataque. Atualmente, o principal vetor de ataque para exfiltração de dados é conhecido como Advanced Persistent Threat (APT).

Os autores [1] definiram o Ciclo de Vida da Exfiltração de Dados - Ciclo de Vida de Exfiltração de Dados (CVED) - em 6 etapas: *Delivery*, *Exploitation*, *Command & Control Channel*, *Exploration*, *Concealment* e *Multistage*. O primeiro estágio é de reconhecimento, onde os atacantes devem analisar seu alvo e identificar vulnerabilidades em seu sistema. Após identificado alguma vulnerabilidade, os atacantes devem criar alguma arma direcionada, por exemplo um *Malware* de acesso remoto, ou um e-mail de *Phishing*. Nos demais estágios serão empregados diferentes vetores de ataque. Os autores ainda ressaltam que dois dos principais vetores de ataque, de múltiplos estágios definidos no ciclo CVED, são: *Insider Attack* e Advanced Persistent Threat (APT). No ataque interno um funcionário, mesmo da empresa ou parceiro, vaza as informações confidenciais para ganhos pessoais.

Já APT é uma classe de ataque cibernético caracterizada por uma ameaça avançada, com um movimento lento e baixo de um grupo de hackers, com objetivo de se manter o maior tempo possível dentro da rede sem ser detectado para prejudicar o alvo ou roubar informações. [6] divide o APT entre 5 etapas: Reconhecimento, estabelecimento de ponto de apoio, movimento lateral, exfiltração/impedimento e pós-exfiltração/pós-impedimento, e ressaltam as diferenças entre ataques tradicionais e ataques de APT, figura 1.1.

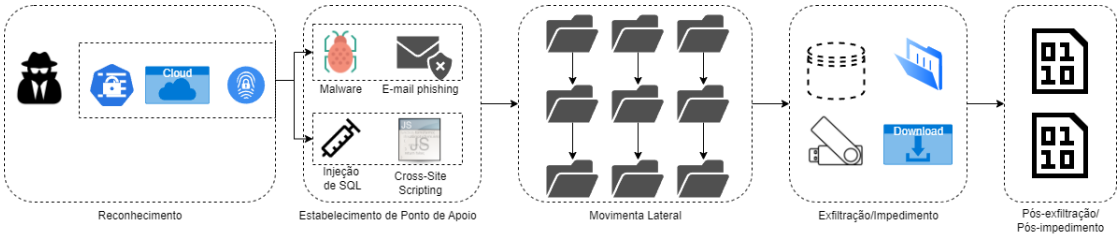

Figura 1.1: *MFog*: Ciclo de Vida de Exfiltração de Dados (CVED) [1]

Devido ao grande desbalanceamento de dados característico do problema de exfiltração, onde grandes aplicações geram milhões de fluxos de dados em um único dia. Comumente a esmagadora maioria desses dados em trânsito são benignos, enquanto uma parte ínfima desses dados são de exfiltração de dados. Neste contexto, algoritmos de inteligência artificial, como: Once Learning (OL), Few-Shot Learning (FSL), One-Shot Learning (OSL), *reservoir computing* e *differential learning* podem ser ideais para resolver o problema, uma vez que são capazes de aprender as características dos dados com poucos exemplos e classes desbalanceadas.

A capacidade humana de rápido aprendizado sempre intrigou os cientistas. Afinal, normalmente, modelos de Aprendizagem de Máquina (AM) em geral necessitam de milhares, senão dezenas de milhares de exemplos para conseguir efetivamente cumprir seu objetivo. Em [9] é relatada uma regra prática bem conhecida: O número de exemplos de treinamento deve ser até várias vezes o número de parâmetros dos objetos a serem classificados, consequentemente os conjuntos de treinamento são extremamente grandes. E, ainda, os dados de treinamento precisam ser classificados, tratados, e em alguns casos até organizados. Portanto, pesquisadores buscaram criar métodos que consigam utilizar um conhecimento previamente adquirido, similarmente ao cérebro humano, e que sejam capazes de aprender com apenas um, ou alguns, exemplos de dados.

1.1 Motivação e defesa cibernética

Nos últimos anos, o Fórum Econômico Mundial (FEM), aponta para a intensificação do uso da cibernética nas cadeias globais de produção – a chamada "Indústria 4.0" –, e

qualificou os ataques cibernéticos como uma das cinco mais prováveis ameaças globais. Organizações e governos tentam se blindar contra ataques cibernéticos, tanto nos aspectos tecnológicos, quanto nos de governança (vide Organização do Tratado do Atlântico Norte (OTAN) implantado em 2008 o Centro de Excelência em Defesa Cibernética Cooperativa (CEDCC)). Os Estados Unidas da América (EUA) aprovaram lei em que o ataque cibernético externo passa a ser reconhecido como "ato de guerra", e a Rússia, que testou com sucesso a desconexão da sua Internet doméstica à rede mundial de computadores, com base na nova de "Lei da Soberania na Internet". No Brasil, a versão de 2008 da Estratégia Nacional de Defesa definiu o setor cibernético como um dos eixos prioritários para a Defesa Nacional. Criado em 2010, o Centro de Defesa Cibernética teve o seu batismo de fogo dois anos depois durante a Conferência das Nações Unidas sobre Desenvolvimento Sustentável (Rio +20). Ações nas tradicionais dimensões – naval, terrestre e aeroespacial – não são mais suficientes para assegurar a defesa dos interesses e da soberania de um país. Como os fatos acima demonstram, o Ciberespaço está consolidado como a mais contemporânea arena para segurança nacional.

Todavia, cabe ressaltar que uma abordagem superficial e romântica, onde a inovação espontânea do mercado suprirá uma nação dos instrumentos necessários a fazer face aos desafios do Ciberespaço, não converge com a realidade. Também, o aparato do setor público, com sua rigidez burocrática e restrições orçamentárias, tende a não ser mais individualmente suficiente para o suprimento de todo o aparato, na velocidade exigida pelo tema. Assim sendo, face à sociedade do conhecimento e visando romper a estanqueidade setorial que não consagrará resultados, há que se ter uma nova dinâmica, com base no modelo original de Tripla Hélice [10], onde as fronteiras entre os setores público, privado e acadêmico se esvaem, dando espaço a ecossistemas onde múltiplas interações entre essas partes se sobrepõem.

Atualmente, os ataques associados à engenharia social são diversos, incluindo as chamadas Advanced Persistent Threat (APT). Estes têm sido objeto de inúmeras investigações; no entanto, ataques cibernéticos de natureza semelhante ao aliciamento foram excluídos desses estudos. Um estudo de *grooming* de [11] é formalizado como um ataque de engenharia social, contrastando seus estágios ou fases com ciclos de vida associados a APT.

Usando o poder de computação do banco de dados gráfico, é possível identificar o invasor e, além disso, é possível detectar outros componentes do sistema afetados. O trabalho do [12] reduziu significativamente o tempo de detecção de violações com essa abordagem e reage rapidamente a novos vetores de ataque. Do [5] desenvolveram a detecção de ataques APT com base em técnicas de análise de rede de fluxo usando aprendizado profundo. Em sua pesquisa, o tráfego de rede é analisado em fluxos de rede baseados em IP e os modelos

de aprendizado profundo são usados para extrair recursos para detectar IPs de ataque APT de outros IPs. Um modelo de aprendizado profundo combinado usando Bidirectional Long Short-Term Memory (BiLSTM) e Graph Convolutional Networks (GCN) é apresentado. Os resultados experimentais mostram que o modelo BiLSTM-GCN tem o melhor desempenho em todas as avaliações de avaliação. Sua pesquisa não apenas mostra que a aplicação de aprendizado profundo na análise de rede de fluxo para detectar ataques APT é uma boa decisão, mas também sugere uma nova direção para técnicas de detecção de intrusão de re de baseadas em aprendizado profundo.

1.2 Objetivos

Com o avanço da Internet das coisas, Internet of Things (IoT), e com a digitalização da informação, cada vez mais diversas empresas utilizamos ambientes virtualizados, em núvem, para armazenamento de seus dados, e de dados de seus clientes. Naturalmente, é de suma importância que essas dados sejam armazenados de forma segura, e que não ocorra roubo de informações. Os artigos [13, 14, 15] são apenas alguns dos que trabalham com segurança no ambiente Hadoop Distributed File System (HDFS), um dos mais utilizamos atualmente. Portanto, é evidente que é preciso mais estudo e que novas técnicas sejam desenvolvidas para solução do problema. Vazamento de dados representam perdas financeiras gigantescas para grandes empresas, e ainda mais, risco de segurança nacional para países. Portanto, a pesquisa desenvolvida neste trabalho visa estudar, e propor, modelos de inteligência artificial para confecção de um sistema de Data Lacking Protection System (DLPS), capaz de detectar tráfico malicioso, relacionado a exfiltração de dados. Nesse sentido, os modelos aqui propostos visam identificar a comunicação maliciosa entre dois sistemas, antes mesmo que esse dado consiga atingir seu objetivo.

Os objetivos específicos deste trabalho são:

- Estudo do estado da arte de Aprendizagem de Máquina (AM) (Aprendizagem de máquina) no combate a exfiltração de dados;
- Estudo de modelos de AM aplicadas a cibersegurança;
- Propor modelos eficazes para o problema;
- Validar os modelos em um banco de dados conhecido e através das técnicas de Meta-Learning (MetL);
- Fazer testes inicias com banco de dados mínimo de caso de exfiltração de dados mapeado.

1.3 Organização do trabalho

Este trabalho está organizado da seguinte forma, o capítulo 2 trata da revisão da literatura na área de Data Lacking Protection System (DLPS), Aprendizagem de Máquina (AM) e Inteligência Artificial (IA), com o foco em Advanced Persistent Threat (APT). Também apresenta o estado da arte dos modelos de Redes Densas, *Transformers* e Redes Robustas. E suas aplicações na cibersegurança. O capítulo também apresenta técnicas de treinamento, desenvolvimento e validação das soluções propostas. Inicialmente é abordado o uso do MetL, Algoritmo Genético (AG) e *AutoML*. O Capítulo também discorre sobre o Few-Shot Learning (FSL), *entropia* e as métricas de validação utilizadas neste trabalho.

O próximo capítulo, 4, trata da metodologia aplicada na confecção deste trabalho. Nesta seção, são dispostas informações sobre a pesquisa e motivação do trabalho, 4.1, confecção, 4.2 e banco de dados de referência, 4.3.

O capítulo 5 discorre sobre a caracterização de um ataque de Advanced Persistent Threat (APT) no domínio da exfiltração de dados, características, sofisticação e desafios relacionados a detecção deste tipo de ataque.

O capítulo 6 apresenta as implementações desenvolvidas neste trabalho, modelos confeccionados e soluções aplicadas. Também discorre sobre suas aplicações, e trabalho necessário para adequação ao problema enfrentando nesta pesquisa.

O penúltimo capítulo, 7, traz os resultados obtidos a partir das soluções propostas, e compara com os resultados utilizados como base da base de dados. Também apresenta as vantagens e desvantagens de cada modelo e solução proposta.

E o capítulo final 8 lista as contribuições realizadas por este trabalho, bem como os desafios e propostas para os trabalhos futuros.

Capítulo 2

Revisão da Literatura

O primeiro passo para confecção de uma nova solução para detecção da exfiltração de dados é entender o estado da arte atual desse campo. Em diversos artigos, pesquisadores identificaram vários problemas para detecção/prevenção. Assim, a literatura para exfiltração de dados tem sido estudada sob diversas perspectivas. A figura 2.1 apresente o presente foco na revisão da literatura realizada por este trabalho. Em [16] os autores destacam os desafios relacionados aos métodos existentes devido a criptografia dos dados, falta de abordagem colaborativa entre a indústria e controles de acessos fracos. Os autores de [17] destacaram canais para exfiltração em ambientes de nuvem. Em [18] os autores identificaram diversos desafios, muitos deles relacionados ao fator humano, para a exfiltração de dados. Em [1] os autores fizeram uma extensa revisão da literatura no campo da cibersegurança, do ponto de vista da IA e Aprendizagem de Máquina (AM).

O presente trabalho visa entender o estado da arte atual no ramo da cibersegurança, e ainda, do ponto de vista do AM, propor a confecção de uma solução dentro do ramo da IA para detecção e prevenção. Assim, sendo essa uma tarefa extremamente complexa, com um número de casos esmagadoramente desbalanceados, tornou-se necessário a adoção de modelos de Redes Neurais (RN) que pudessem aprender com um conjunto de testes extremamente curto. Portanto, modelos de Meta-Learning (MetL) são ideias para o problema.

2.1 Aprendizagem de Máquina (AM) e Aprendizagem Profunda (AP)

O Once Learning (OL) é uma classe de algoritmos que tentam aprender os dados apenas com um ou alguns exemplos, [19] foi o primeiro modelo que conhecemos voltado para algoritmos que aprendem apenas com poucos dados. A proposta em [19] é uma rede

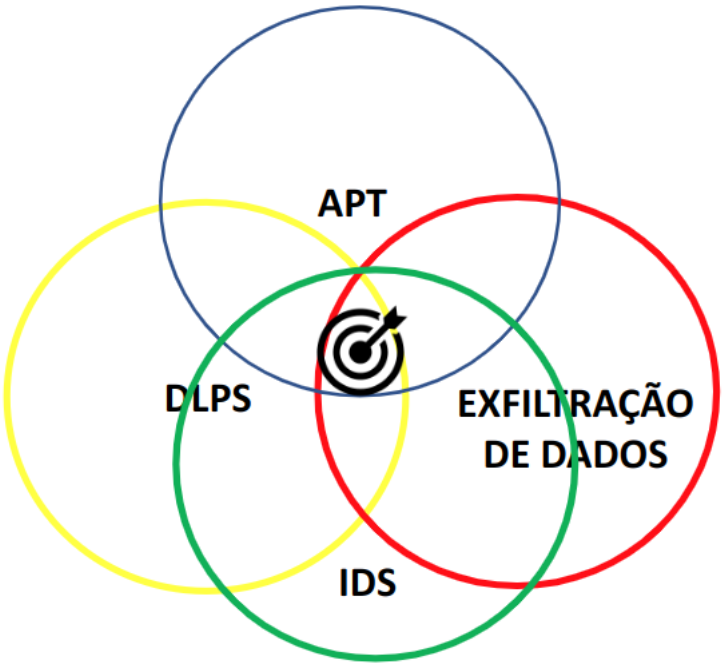

Figura 2.1: Revisão da literatura

neural de mapas paralelos, com duas matrizes paralelas, uma chamada de neurônios pré-sinápticos e outra chamada de neurônios pós-sinápticos. Nesse sentido, para cada entrada, é encontrada a representação direta mais adequada para aquele dado em um dos neurônios da rede, e em seguida é realizado o processo de inferência pós-sináptico. A rede possui alta capacidade de paralelismo, e foi pensada de maneira a ser adequada à computação quântica.

Após o trabalho de [19] surge o One-Shot Learning (OSL) - proposto por [9] o qual é um modelo de aprendizagem rápida, e de rápida convergência, para classificação de dados dentro da área da visão computacional. Normalmente, modelos de ML necessitam de milhares de dados para sua convergência, enquanto o OSL visa aprender informações sobre os objetos com um, ou apenas alguns exemplos, normalmente de imagens. Em [9] os autores utilizam uma versão do OSL chamada de Bayesian One-Shot Learning (BOSL) para categorização de objetos. Nele são utilizados para treinamento do modelo apenas de 1 até 5 imagens. Assim, os autores conseguem utilizar um métodos *Baseyan* para permitir que o modelo incorpore um conhecimento previamente adquirido por meio de um função de densidade de probabilidade.

Em [20], os autores descrevem Few-Shot Learning (FSL) como o processo de treinamento de um modelo de ML com pouquíssimos dados de treinamento. Ele é baseado na capacidade humana de generalizar novos conhecimentos com base em exemplos previamente memorizados.

Diferente desse processo, modelos de IA normalmente necessitam de milhares, senão dezenas de milhares, de exemplos para convergirem em um modelo adequado. Logo, pesquisadores objetivaram-se na criação de um novo modelo de AM que fosse capaz de simular essa capacidade humana, e pudesse aprender novas categorias de dados de forma rápida e com um número pequeno de amostras. Assim, para uma rápida convergência, e para o uso de variadas categorias, essa abordagem utiliza um conhecimento previamente adquirido para melhorar o desempenho do modelo. Portanto, o algoritmo de FSL é um dos mais adequados para o problema abordado por esse trabalho, já que, o número de *logs* importantes para detecção é infinitamente menor do que os *logs* comuns.

As Redes Diferenciais (RD) - são redes que utilizam as redes neurais para aprenderem a solução de equações diferenciais, e por esse motivo podem ser envoltas nas técnicas de *Lyapunov* para medir a sua capacidade de convergência. Pela característica de aprenderem a variação dos dados, são adequadas para rápida convergência em tempo real e por esse motivo podem ser adequadas para o problema enfrentado. Os autores de [3] em seu trabalho generaliza as RDD para diversas camadas, com provas de convergência utilizando as técnicas de Lyapunov e aplica as redes ao problema de sinais cerebrais.

[21] propõe uma nova divisão dos subcampos de estudos para a IA, classificando-as entre: Redes neurais inspiradas em humanos, redes neurais inspiradas em biologia, redes neurais de máquina e redes neurais quânticas. Este trabalho irá desenvolver redes neurais de máquinas.

Já o Meta-Learning (MetL) é um subcampo do Aprendizagem de Máquina (AM) que busca aprender a aprender, ou seja, é um conjunto de técnicas para a análise do processo de aprendizado dos algoritmos de inteligência artificial capaz de melhorar a performance dos algoritmos e a sua velocidade de convergência na tarefa atual ou em novas tarefas nunca antes vistas. Além disso, o MetL pode utilizar os próprios algoritmos de inteligência artificial para aprender o processo de aprendizado de um segundo algoritmo, onde existem diversas abordagens de sucesso com essa característica. [22] define MetL como a ciência de sistematicamente observar como diferentes abordagens de AM se saem em uma grande variedade de tarefas de aprendizado, e então aprender dessa experiência, ou meta-dados, a aprender novas tarefas muito mais rápido do que seria possível de outras maneiras. Hospedales, 2020, define MetL como o processo de explorar a experiência de múltiplos episódios de aprendizado - geralmente com uma distribuição de tarefas correlatas - e usar essa experiência para melhorar a performance de aprendizado futuro.

No contexto da segurança cibernética, MetL também pode ser aplicado, uma vez que em contextos em que existem adversários, rapidamente a situação e comportamentos mudam e os algoritmos e técnicas utilizados já não são mais eficientes para resolver um determinado problema. Um meta-modelo no contexto do MetL, segundo [22], é um modelo

capaz de aprender a propor novas estruturas de redes neurais capazes de resolver uma nova tarefa nunca antes vista. Nesse sentido, o MetL pode resolver o problema do adversário por meio do treinamento de redes capazes de criar novas entidades capazes de contrapor as suas novas manobras. [23] propõe uma rede de MetL capaz de gerar novos algoritmos mesmo em dados com corrupção adversária. [24] propõe uma rede de MetL com FSL capaz de ser robusta mesmo em casos de corrupção de dados, uma vez que segundo o autor as técnicas que utilizam FSL são muito frágeis à esse tipo de ataque.

A Computação por Reservatório (CR) ou computação por reservatórios é um subcampo da computação que tem entrelaçamentos fortes com o campo de ML. As redes neurais podem ser construídas a partir da teoria básica da computação por reservatórios, e muitos mecanismos cerebrais já foram explicados a partir desse ponto de vista [25]. Esse autor define computação por reservatórios como um processo que busca aproximar uma função com entradas e saídas, com um mecanismo interno complexo como uma rede neural. As redes neurais da computação por reservatórios possuem reservatórios, que são partes não alteradas durante o processo de treinamento, as quais são utilizadas externamente por um interpretador, o qual é modificado. Este processo pode gerar uma convergência mais rápida do que o processo tradicional do Aprendizagem Profunda (AP) uma vez que menos parâmetros são testados.

Além disso, a computação por reservatórios possui o grande desafio de integrar os sistemas de maneira mais generalista. Devido à sua característica de construir redes aleatórias para a realização da aproximação da função, nós cogitamos que obter sucesso nessa tarefa significa se aproximar de uma inteligência artificial geral, ou seja, uma inteligência artificial capaz de resolver diversos problemas diferentes. Uma vez que esta característica é muito importante no contexto cibernético do processo de identificar APT, onde existe um adversário, e pela capacidade de rápida convergência desse modelo, acreditamos que acrescentar e comparar esse modelo aos outros possa trazer grandes frutos no futuro no sentido de uma inteligência geral para a cibernética, e agora uma solução para exfiltração de dados em trânsito.

2.2 Reconhecimento de atividade humana e dados fora da distribuição

O campo de cibersegurança envolve diversas tarefas de Reconhecimento de Atividade Humana (RAH). No contexto da detecção de exfiltração de dados trata-se da detecção de atividades de seres humanos ou robôs na rede, de maneira bem intencionada ou mal intencionada. Por esse motivo, diversas soluções provenientes do RAH podem ser utilizadas também para a detecção da exfiltração de dados.

[26] entende o campo de reconhecimento de atividades humanas como a tarefa de dado um determinado contexto e conjunto de dados, verificar qual atividade um ser humano está fazendo, ou mesmo se aquela é uma atividade normal a ser realizada. Um grande exemplo de contribuição dada pelo RAH ao campo da cibersegurança se encontra no trabalho [27] aonde se tentar identificar atividades não humanas em um determinado contexto.

Para o problema da exfiltração de dados, deve-se identificar atividades humanas e não humanas, mas também identificar a intenção dessas atividades. De certa maneira, entender a própria exfiltração de dados como uma atividade, ou mesmo algo inesperado como Dados Fora de Distribuição (DFD), enquanto um envio bem intencionado de dados pode ser considerado outra atividade. Dessa forma, conhecer a performance e soluções em problemas de RAH pode ser uma boa maneira de validar as técnicas que aplicadas neste trabalho. Uma vez que, não existem bancos de dados relacionados a exfiltração de dados, por meio do fluxo de pacotes TCP/IP, disponíveis na literatura.

2.3 Redes densas

Redes Diferenciais (RD) são redes profundas com múltiplas camadas altamente conectadas. O modelo é composto por camadas conectadas onde cada camada recebe conexões da camada imediatamente anterior a ela, e também fornece conexões a camada seguinte, [28]. A figura 2.2 mostra um modelo de exemplo de RD, é possível observar a densidade das conexões de uma camada entre as camadas imediatamente anterior e seguinte. Neste modelo, a camada de saída possuí apenas um nó, resultando em uma saída binária, e uma função *softmax*. A entrada, no caso do modelo desenvolvido neste trabalho, é um vetor de atributos extraídos do pacote de dados capturado.

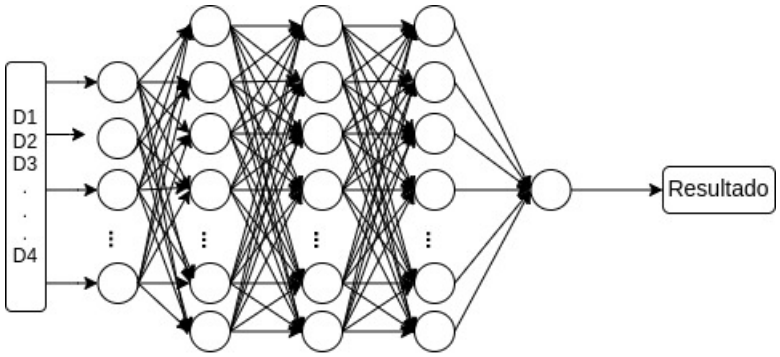

Figura 2.2: Modelo de DNN de exemplo

Em [29] o autor explica que as várias camadas das Redes Diferenciais (RD) podem ser usadas como camadas de abstrações. Assim, cada camada do modelo pode identificar

diferentes características do problema em relação as demais. Portanto, essas múltiplas camadas dão às redes profundas uma vantagem conveniente para aprendizagem de problemas complexos. Um problema bem conhecido na literatura, já que a profundidade das redes neurais permitiu aos perceptrons a solução de problemas não lineares já na década de 50.

2.4 O transformer

O modelo *transformer* foi proposto pela primeira vez em 2017 em um artigo chamado "atenção é tudo o que você precisa"[2]. O modelo *transformer* pode ser usado em sequências temporais ou para relacionar um dado com mais de uma variável. A sua grande vantagem é sua capacidade de paralelizar com complexidade $O(1)$, enquanto outras redes como as redes Redes Recorrentes (RR) precisam de $O(n)$ passos sequenciais. Além disso, o *transformer* tem uma capacidade infinita para relacionar dados temporais, mesmo dados separados por grandes distâncias temporais, e pode prestar atenção nas mais relevantes relações entre os dados.

Podemos resumir o modelo matemático do *transformer* como se segue [30]. Sendo estas as equações que formam uma camada "*transformer*"as quais podem ser empilhadas várias vezes em profundidade.

$$\begin{aligned} Q^h(x_i) &= W_{h,q}^T x_i \\ K^h(x_i) &= W_{h,k}^T x_i \\ V^h(x_i) &= W_{h,v}^T x_i \\ W_{h,q}^T, W_{h,k}^T, W_{h,v}^T &\in R^{d \times k} \end{aligned} \tag{2.1}$$

Com x_i sendo a *timestamp* i da sequência, h indicando o tamanho de cada cabeça, q, k e v indicando o tamanho da Q *query*, da K *key* e do valor V. Os quais serão explicados mais abaixo. O índice i indica qual é a entrada, uma vez que cada modelo de *transformer* possui uma quantidade específica de cabeças escolhida pelo usuário, com as quais as séries de entrada serão divididas, para se fazer diferentes análises e relacionamentos, ou seja, cada cabeça indica uma parte do dado que será analisada como uma unidade por determinados mecanismos de atenção. T indica a operação de transposição.

$$
\begin{aligned}
\alpha_{i,j}^{(h)} &= softmax_i(\frac{< Q^{(h)}(x_i), K^{(h)}(x_j) >}{k^{(1/2)}}) \\
u_i^* &= \sum_{h=1}^{H} W_{c,h}^T \sum_{j=1}^{n} \alpha_{i,j}^{(h)} V^h(x_j) \\
z_i^* &= W_2^T ReLU(W_1^T u_i) \\
z_i &= LayerNorm(u_i + z_i^*; \gamma_2; \beta_2)
\end{aligned}
\tag{2.2}
$$

softmax é uma função utilizada para gerar uma probabilidade entre 0 e 1, neste caso, ela está sendo utilizada para responder às perguntas: quão relevante é essa informação? O quanto o modelo deve prestar atenção a essa informação? O valor alfa α é justamente o resultado desta operação, para cada *head* e para cada valor de cada *head*, sendo portanto uma matriz. Assim, a matriz u é a junção de todas as cabeças multiplicadas pelos pesos relativos de cada valor, unificando dessa forma, os dados em uma matriz, que em seguida é utilizada como entrada para uma camada densa (uma rede densa com uma camada escondida e uma função de ativação), a qual é ainda normalizada para manter um certo padrão nos dados resultando em z.

Aonde Q, K e V são a $Query$ (busca), Key (chave) e Valor (V) respectivamente. Nesse sentido, o *transformer* vai tentar combinar diferentes *timestamps* para descobrir quais são as mais relevantes para minimizar o erro. Destarte, são construídos 3 espaços vetoriais diferentes: O da busca, aonde se faz uma "busca" sobressaltando-se valores interessantes para o modelo, de um ponto de vista matemático. O segundo espaço vetorial é a *key*, serão os valores observados para definir se essa busca retornou um resultado razoável ou não. Estes dois espaços vetoriais são relacionados pela função *softmax*, que vai definir uma probabilidade de aquele valor ser relevante, ou não, a partir da chave respectiva. Por último, temos o valor concreto, que é aquele valor cuja rede utiliza para representar e entender a informação, a passando para frente nas operações de *feed-foward*, ou "synapses artificiais". O valor é multiplicado pela probabilidade resultante do *softmax*, e pode-se entender tal multiplicação como: Quanto desse valor é relevante para a minha decisão, qual abstração a rede deverá fazer, ou mais reconhecidamente, para aonde a rede deverá prestar atenção.

Uma explicação mais detalhada sobre esse modelo pode ser encontrada no artigo "atenção é tudo o que você precisa" [2] em conjunto com o artigo [30]. Em uma breve explicação, o *transformer* funciona com um codificador e um decodificador. Para entender dados temporais o *transformer* adicionar uma camada posicional, a qual foi explicada anteriormente, que é responsável por somar informações sobre posição temporal na entrada. A entrada é uma sequência, por exemplo, nove momentos das ações em uma bolsa de valores. A camada *multi-head-attention* é responsável por prestar atenção às mais

importantes informações e relações entre as sequências. No decodificador, em um modelo sequência-para-sequência a saída gerada anteriormente é adicionada como uma entrada, isso é para, por exemplo em problemas textuais, o *transformer* prestar atenção no que ele já escreveu, e entregar saídas consistentes com as previsões anteriores.

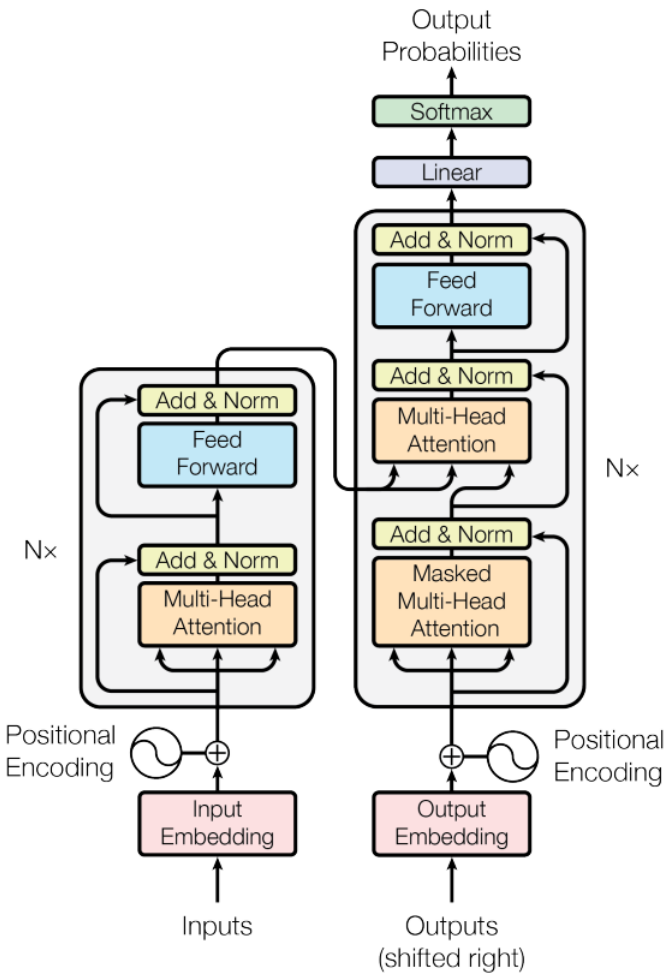

Figura 2.3: Arquitetura do *transformer, Attention Is all You Need* [2]

Neste trabalho a entrada do decodificador será a mesma da do codificador. Vamos simplesmente adicionar a mesma informação para ajudar o decodificador a entender a representação criada pelo codificador.

2.5 Redes robustas

As redes robustas são modelos inventados para garantir, sob determinadas condições, que um modelo não irá explodir e irá convergir para um mínimo global. Ao contrário dos

modelos heurísticos, as redes robustas não utilizam algoritmos como a *backpropagation* para a evolução dos pesos do modelo. Na verdade, os pesos do modelo são calculados a partir das técnicas de estabilidade de *Lyapunov* [31]. Nesse processo, uma função de distância, positivo definida $V(X) > 0, paratodoX$ é utilizada para averiguar se os pesos irão permanecer dentro de uma zona de convergência.

Nesse caso, as redes são chamadas de redes diferenciais, pois a rede está inserida em uma modelagem de equação diferencial, e é responsável por aproximar a solução da equação diferencial. Podemos encontrar várias aplicações dessas redes como no artigo [3].

A estratégia consiste em encontrar as leis de adaptação corretas para um determinado modelo conseguir fazer com que a derivada da função de distância seja sempre negativa, a partir de um certo tempo t, considerando um sistema de equações diferenciais. Nesse sentido, as redes robustas são em sua maioria modelos equivalentes aos modelos de Redes Recorrentes (RR) utilizando uma rede densa e um vetor de estados para prever a trajetória de um sistema estruturado como uma equação diferencial. Podemos mapear o sistema de equações diferenciais como se segue:

$$X' = W^* S(V^* X)) + Dt \tag{2.3}$$

Onde X' é a derivada (ou menor variação) da variável de estados do sistema, $W^* S(V^* X)$ é o regressor, equivalente a uma rede densa com uma camada escondida, onde W^* e V^* são os pesos ideais (a melhor solução) e Dt é o erro mínimo possível para a previsão do funcionamento do sistema. É importante ressaltar que os pesos $W*$ e $V*$ ideais não são conhecidos, mas são usados matematicamente para encontrar as melhores leis de adaptação, ou seja, encontrar a forma de atualizar os pesos da rede neural de forma que ela encontre os melhores resultados possível sem divergir.

Dessa forma, podemos definir a função de distância, ou candidata à função de *lyapunov*, como sendo:

$$V = tr(W' - W^*) + tr(V' - V^*) + E^2 \tag{2.4}$$

Onde W' e V' são os valores estimados, e E é o erro de estimação. O erro de estimação pode ser definido de diversas formas. Geralmente as leis robustas utilizam o próprio y como parâmetro para o cálculo do erro, com uma subtração entre a matriz atual e a matriz perfeita, para cada camada. como:

$$E = (X - X^{'})^2 + (W^* - W^{'})^2 + (V^* - V^{'})^2 \tag{2.5}$$

Onde X é a própria variável do sistema, leia-se um vetor, X' é a variável estimada, $W*$ e $V*$ são os parâmetros perfeitos para cada camada, e W' e V' os parâmetros estimados.

Obviamente os parâmetros perfeitos não são conhecidos, porém, para o problema ser válido, precisamos considerá-los. Nesse sentido, a função de distância permite, com algumas manipulações de dominação, achar parâmetros adequados para manter a derivada da distância sempre inferior a zero, e conseguir um sistema que sempre converge para o mínimo global para entradas limitadas. Esse processo de dominação é o que determina qual será a composição das leis de adaptação de cada camada da rede. Aqui está, também, uma das limitações do estado da arte das redes robustas, ainda não foram encontradas opções de leis de adaptações gerais para um número qualquer de camadas, limitando o tamanho das redes para àqueles aos quais podemos encontrar as leis de adaptação matematicamente.

Em seguida, após definir o erro, tomamos a derivada do erro e calculamos as leis de adaptação para W' e V' (uma para cada) de forma que $V^{'} < 0$ e $W^{'} < 0$, o que garante a robustez do sistema, assumindo que todas as entradas são limitadas. Assim chegamos às seguintes leis de adaptação utilizadas neste projeto:

$$\begin{aligned} V^{'} = V^{'} - Y_w * (a_w * ||E||_{fro} * (V - V_0) \\ + W * s(V * Z) * W * B * (K * E)^T * Z) \end{aligned} \tag{2.6}$$

$$\begin{aligned} W^{'} = W^{'} - Y_w * (a_w * ||E||_{fro} * (W - W_0) \\ + B * K * E^T * s(V * Z) - B * (K * E)^T * W * s(V * Z) * (V * Z)) \end{aligned} \tag{2.7}$$

Onde Y_w, a_w são constantes escolhidas pelo usuário. Tal como A, B, K são matrizes escolhidas pelo usuário com o mesmo número de neurônios da camada de entrada. $||E||_{fro}$ representa a aplicação da norma de Frobênios. Aqui E não é o erro definido anteriormente, mas um simples vetor resultante da subtração do valor predito, e do valor real. S é o regressor utilizado, sendo normalmente uma função senoidal. Essas constantes e matrizes definidas pelo usuário surgem de manipulações matemáticas inseridas no meio do processo do cálculo de dominação das leis de adaptação. Essas constantes e matrizes podem ser utilizadas para ajustar a velocidade de convergência e a precisão do modelo, conforme o que é observado experimentalmente para cada dado, e as limitações das tecnologias empregadas para implementar a rede. Muitas vezes escolher um valor ruim para essas constantes, pode levar a explosões, devido às próprias limitações do sistema imperfeito empregado para implementar o modelo.

2.5.1 Classificação com redes robustas de sinais temporais

Para utilizar as redes robustas como classificadores de sinais temporais, existe uma técnica [3] em que para cada classe, existirá um classificador. Ou seja, se existe um problema com duas classes, treinaremos uma rede robusta para identificar cada uma das classes

(ou prever o comportamento de um sinal de cada classe). Para fazer a classificação, rodamos paralelamente cada um dos classificadores sobre o mesmo sinal, e calculamos o erro quadrático médio de cada classificador. Aquele que obtiver o menor erro, definirá a classe do sinal, ou seja, o sinal se pareceu mais com aquele tipo de classe do que com todas os outros tipos.

Formalizando, temos, para um problema de duas classes:

$$S^{(1)} = W^{(1)}S(V^{(1)}X) \tag{2.8}$$

$$S^{(2)} = W^{(2)}S(V^{(2)}X) \tag{2.9}$$

Sendo (1) o classificador, ou regressor, da classe 1 e (2) o classificador ou regressor da classe 2. Em seguida, para um determinado sinal temos:

$$E^{(1)} = \sum_{j=1}^{n}(X_j'^{(1)} - X_j)^2 \tag{2.10}$$

$$E^{(2)} = \sum_{j=1}^{n}(X_j'^{(2)} - X_j)^2 \tag{2.11}$$

$$E^{(1)} = E^{(1)}/n \tag{2.12}$$

$$E^{(2)} = E^{(2)}/n \tag{2.13}$$

Aonde E é o mesmo erro definido na equação 2.7 e X' é o valor previsto por cada modelo, sendo eles tantos quanto forem as classes existentes e X é o próprio valor do sistema. Assim, a classe pode ser definida como:

$$classe = index\_min(E^{(1)}, E^{(2)}) \tag{2.14}$$

Onde classe é a previsão do sistema, relativa ao índice i do modelo, como cada modelo representa uma classe e possui um índice, é procurado o menor erro e o modelo detentor deste, indicará à qual classe a série pertence.

Ou seja, o regressor que tiver o menor erro médio com relação à previsão de uma série a ser classificada, responderá à qual classe essa série pertence. Isso decorre pelo fato de que cada regressor pode apenas prever um tipo de série. Como podemos ver, a grande desvantagem dessa abordagem é que para cada classe, deverá ser adicionado um novo regressor, o que pode demandar altos custos computacionais. O que mitiga esse problema é o fato de geralmente as redes serem bem pequenas, mas eficientes, devido às limitações

explicitadas na sessão anterior. Além disso, as redes diferenciais são concebidas na maior parte das vezes em tempo real, o que será o caso de sua aplicação neste trabalho.

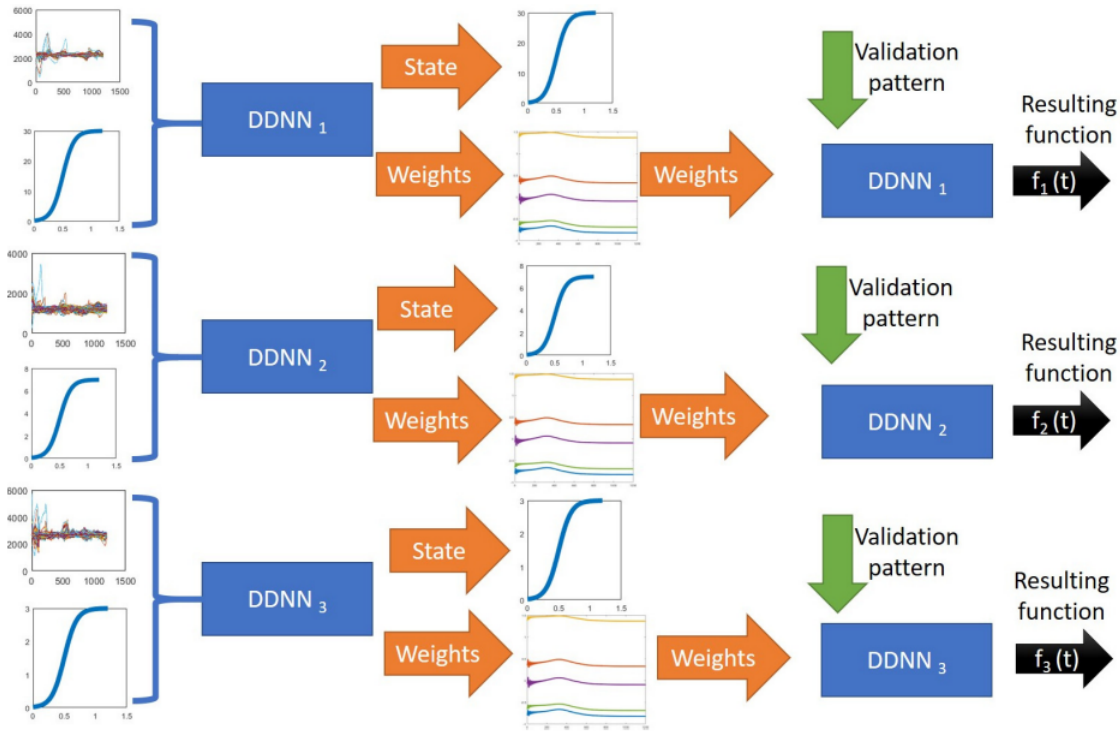

Figura 2.4: Esquema de classificação com redes robustas [3]

Na imagem é possível perceber 3 redes robustas (Redes Diferenciais Densas (RDD)), cada uma roda em cima da mesma série temporal, em seguida, gera um determinado "padrão" que é utilizado em seguida para classificar a série de entrada, a partir de um padrão de validação (*Validation Pattern*), comparando a função de saída f, com o padrão esperado. Essa validação é justamente, nesse caso, o cálculo do erro com relação à própria série visualizada, caso o modelo aproxime bem a série, provavelmente a classe da série é aquela respectiva ao modelo.

Apesar de problemas com exfiltração de dados não serem novos, ano após anos as técnicas utilizadas para roubo de informações se aperfeiçoam. Portanto, qualquer técnica aplicada ao seu combate deve ser extremamente eficaz, e trabalhar bem com baixa quantidade de dados disponíveis, ou grande capacidade de adaptação. Este capítulo irá apresentar as técnicas aplicadas na confecção deste trabalho. E está organizado da seguinte forma, a seção 3.1 abordará o uso do Meta-Learning (MetL), Algoritmo Genético (AG) 3.1.1 E *autoML* 3.1.2. A seção 3.2 discorre sobre o Few-Shot Learning (FSL). A

seção 3.3 apresenta o uso e cálculo da Entropia. E a seção 3.4 aborda as métricas de validações aplicadas.

2.6 Meta-Learning (MetL)

O Processo de Meta-Learning (MetL) é o processo de aprender a aprender. Nesse sentido, a ciência do MetL foca em utilizar o conhecimento sobre tarefas anteriores, para aprender a como realizar novas tarefas. Dessa forma, digamos que exista um banco de dados com diversas tarefas resolvidas por inteligências artificiais no contexto da cybersegurança e da exfiltração de dados. Porém, neste momento, temos a tarefa de treinar uma nova inteligência artificial para detectar um novo tipo de exfiltração de dados detectado por especialistas. O MetL nos ajudará a utilizar todo esse conhecimento, para construir rapidamente e de maneira mais eficiente essa nova inteligência artificial, com base nos conhecimentos anteriores.

Dessa forma, o MetL se encarrega de encontrar semelhanças entre determinadas tasks, verificar como o processo de aprendizado de uma inteligência artificial vem ocorrendo e intervir de formar que este processo acelere e aumente a sua eficácia. Além disso, podem ser realizadas transferências de conhecimento de uma task para outra task, quando tal tarefa for julgada útil. Por exemplo, um processo de pré-treinamento pode ser considerado uma transferência de conhecimento, pois pode-se utilizar uma parte da rede pré-treinada e aproveitar aquele conhecimento semelhante em outra rede. O MetL indicará quais as melhores maneiras de cumprir essa tarefa, como construir as topologias e qual seria a eficiência dessa topologia.

Em sua grande parte o MetL é composto por algoritmos de aprendizado genético, responsáveis por fazer buscas intensas pelas possibilidades tomando como base o cruzamento de informações e semelhanças entre tasks. Além disso, algoritmos de *autoML* (algoritmos responsáveis por construir arquiteturas de redes neurais) podem ser treinados neste ambiente e realizar tarefas parecidas. Tal como outros algoritmos auxiliares podem utilizar machine learning para aprender a prever a performance de um modelo sem a necessidade de treiná-lo, aprender a prever a complexidade de um determinado problema, dado o histórico, e a calcular o custo benefício de resolvê-lo

Nesse sentido, existem no mercado algumas iniciativas relativas à plataformas para desenvolvimento colaborativo de algoritmos de machine learning. Uma dessas plataformas, contendo funções únicas, é a plataforma de MetL da Panaceia. Escolhemos essa plataforma para fazer o desenvolvimento dos algoritmos, assim, realizamos toda a captura dos meta-dados de treinamento pela plataforma, usamos suas funções de algoritmo genético 3.1.1.

Outras vantagens prometidas para o futuro é o uso do MetL em tarefas multimodais e para transferências de conhecimento, como temos em [32], e cuja será facilitado nesta pesquisa pelo uso de uma plataforma de MetL, mesmo que ela venha a mudar, poderemos manter os meta-dados. Além disso, uma das tarefas que serão citadas na revisão da literatura se referem à construção futura de algoritmos de autoML capazes de gerar novos modelos defensores em um tempo curto, tarefas parecidas também foram realizadas em [33] e serão também facilitadas pelo uso de uma plataforma de MetL, e do MetL em si.

Este projeto fez uso das tecnologias totalmente desenvolvidas pela Panaceia, a plataforma de MetL foi desenvolvida inteiramente no ambiente da Panaceia e não teve relação com o desenvolvimento deste projeto, nem usou recursos oriundos da Universidade de Brasília para sua confecção. A plataforma foi escolhida para ser usada devido às suas tecnologias únicas no mercado e ao fato de os membros da equipe já possuírem facilidade em seu manuseio. o projeto foi construído a partir de uma licença especial fornecida pela Panaceia, permitindo a divulgação científica e o uso das topologias de Inteligência artificial geradas como a Universidade de Brasília e os pesquisadores deste projeto desejarem.

2.6.1 Algoritmo genético

Algoritmos genéticos são técnicas de buscas, derivadas dos algoritmos evolutivos AG. São algoritmos simples, e extremamente eficientes para busca em espaços complexos. Que simulam o processo da seleção natural dos melhores indivíduos. Define por indivíduos possíveis soluções ao problema. E população o conjunto de indivíduos considerados. Em [34] os autores definem o algoritmo genético, AG, como um método de busca com heurística adaptativa baseado em genética de populações, introduzido por John Holland no início dos anos 90. Um algoritmo de busca probabilística baseado na seleção natural. O algoritmo é iniciado com uma população inicial. A cada ciclo, chamado de geração, cada indivíduo, ou solução, é avaliado de acordo com seu *fitness value*, ou desempenho para o problema proposto. E ao final da geração são selecionados probabilisticamente de acordo com sua pontuação. Pode ocorrer também a combinação de indivíduos para produção de novas soluções. A cada nova geração, novos indivíduos melhores são gerados. O algoritmo para ao encontrar uma solução ótima, ou, após um certo número de gerações. O algoritmo 2 define um pseudo-código de um algoritmo genético.

Apesar de se tratar de um algoritmo de busca, AG também são aplicados em técnicas de IA, [35]. Esses algoritmos são extremamente eficientes na busca pela melhor modelagem de um modelo de IA. Neste caso, o indivíduos são modelos de inteligência artificial, gerados de forma aleatória. Ao final de cada geração a pontuação de cada membro da população é calculada pela proximidade de sua solução ao problema, ou seja, sua acurácia. Os melhores indivíduos possuem alta probabilidade de serem selecionados. A maioria dos

Algorithm 1 Pseudo-código do Algoritmo Genético

As formas de entrada e saída são definidas e recebem uma camada de entrada e uma camada de saída densa sem função de ativação. Uma camada oculta de um neurônio é definida.
O número de gerações a serem encontradas é definido.
A função de perda, loss function, a ser usada nas camadas ocultas é definida.
O número máximo de neurônios em uma camada densa é definido.
A paciência para um retorno de chamada é definida.
Os dados a serem treinados são definidos.
while o número de gerações não é atingido **do**
    gerar um san sem alterações da geração atual.
    No san, camadas ocultas densas aleatoriamente podem ser adicionadas ou removidas com 50% de probabilidade.
    No san, o número de neurônios de uma camada oculta é alterado aleatoriamente.
    A rede san é treinada com callback com paciência como a escolhida.
    if San_loss_function < Father_loss_function **then**
        geração_atual = San
        geração_número = geração_número + 1
        send_meta_data()
    else
        geração_atual não é atualizada.
    end if
end while

métodos de seleção são projetados para que uma pequena porção de indivíduos com baixa pontuação sejam selecionados, mantendo a diversidade da população. Alguns dos modelos selecionados são combinados para geração de novos indivíduos, processo conhecido como *crossover*, ou combinação/mutação. Indivíduos gerados a partir desse processo compartilham inúmeras características de seus "pais". Técnicas de AG são aplicadas na área da robótica [36], jogos [37], inteligência artificial [35, 38, 39], processamento de imagens [40], etc.

2.6.2 AutoML

Muito autores não consideram o *autoML* como um campo do Meta-Learning (MetL), no entanto, segundo as definições do MetL decidimos considerar um campo do MetL, conforme diversos outros autores [22]. O *autoML* munido de MetL neste trabalho está sendo preparado a partir da coleta dos meta-dados de arquiteturas de modelos e suas performances sobre as tarefas a que são submetidas. Nesse sentido, gerar um algoritmo de *autoML* consiste em gerar um modelo de inteligência artificial capaz de receber como parâmetros os dados específicos de uma determinada task (ou tarefa) e construir com precisão uma arquitetura de inteligência artificial capaz de executar esta tarefa.

Neste trabalho, o nosso objetivo será construir as pré-condições necessárias para a criação de um *autoML* capaz de construir arquiteturas de inteligência artificial para novos tipos de ataques de exfiltração de dados. Tal tarefa, se alcançada no futuro, permitirá àqueles responsáveis pela proteção de dados uma adaptação rápida e contínua, ainda mais se utilizando plataformas de desenvolvimento de inteligências artificiais como a plataforma de MetL da Panaceia, a qual foi selecionada para a realização deste trabalho, devido às suas características únicas relativas à captura dos meta-dados desejados.

2.7 Few-Shot Learning (FSL)

O Aprendizagem Profunda (AP) é uma das áreas de pesquisa mais populares na atualidade. E foi implementado e adaptado a todas as áreas de domínio da ciência moderna, provando ser extremamente efetivo. Contudo soluções convencionais de modelos de AP necessitam de centenas de milhares de dados classificados, e um alto número de iterações no *dataset* para treinamento [41, 42, 43]. Portanto, tornou-se necessário a criação de modelos capazes de simular a capacidade humana de aprendizagem. Diferentes das técnicas convencionais de AP humanos são extremamente eficientes na identificação de objetos com um número baixo de exemplos. Os autores de [44, 45] propõem um novo paradigma de aprendizagem a partir de bases de conhecimento com número limitado de exemplos, chamado de One-Shot Learning (OSL), ou Few-Shot Learning (FSL).

Em [20] os autores descrevem FSL como o processo de treinamento de um modelo de AM com pouquíssimos dados de treinamento. Ele é baseado na capacidade humana de generalizar novos conhecimentos com base em exemplos previamente memorizados. Comumente, algoritmos de IA necessitam de bases de conhecimento com centenas de milhares de dados. Em [45] os autores expõem que o algoritmo de FSL são adequados em casos onde o baixo número de exemplos são relativos à problemas éticos, de segurança ou privacidade.

Os autores de [41] argumentam que modelos de FSL focam no aprendizado de novas categorias com baixo número de exemplos, e geralmente são combinados com técnicas de MetL, 3.1. Similarmente ao experimento proposto por este trabalho, que utiliza técnicas de FSL e MetL para captura e estudo dos *hyper-parâmetros* dos modelos propostos.

2.8 Entropia

A entropia é uma medida de quantidade de informação no desenvolvimento da teoria da informação de *Claude Shannon*, [46]. *Claude Shannon*, engenheiro e matemático, é considerado pai da teoria da informação. Introduziu no ano de 1948, no trabalho *A*

Mathematical Theory of Communication, conceitos primordiais que originaram a teoria da informação. Entre os conceitos propostos está a **Entropia**. Nesse trabalho, *Shannon* desenvolve um modelo teórico para o problema da comunicação envolvendo a transmissão de mensagens digitais. Para ele, é fundamental a transmissão de uma mensagem remotamente, com um grau de fidelidade desejado, independentemente de seu significado. Segundo ele, o processo de geração de informação é um processo não-determinístico, pois, não é possível determinar a priori o canal onde será transmitido. *Shannon* definiu o cálculo da Entropia na equação 3.1, sendo H a entropia, x um evento dentro do conjunto de possibilidades, N o número de elementos, e $P(x_i)$ a probabilidade de ocorrência de i. Na fórmula é calculado o logaritmo de uma probabilidade, que por definição é um número entre 0 e 1, portanto, o resultado é um número negativo, e portanto é colocado um sinal negativo no início da equação, que serve para tornar o resultado um número positivo.

$$H(x) = -\Sigma_{i=0}^{N-1} P(x_i) \log_2 P(x_i) \tag{2.15}$$

Neste trabalho, o cálculo da entropia foi utilizado para priorização de informações no modelo de redes neurais densas, 2.3. Esta demonstra ser uma abordagem computacionalmente leve. Assim sendo, é adicionado um modo de pré-processamento aos dados de entrada do modelo de DNN, com o cálculo de entropia de IP de origem e destino. Com isso, é possível observar que com o crescimento do número de pacotes com mesmo IP de origem/destino, o valor do cálculo de entropia aumentará proporcionalmente.

2.9 Métricas de validação

Para validar a performance dos resultados obtidos, este trabalho adota as seguintes métricas nas análises temáticas e estudos de casos

- **Confusion Matrix**: por meio da matriz de confusão, calcula-se a quantidade de falsos positivos e falsos negativos, bem como de verdadeiros positivos e verdadeiros negativos, propiciando calcular acurácia e sensibilidade;
- **Classification Report**: relatório com as principais métricas de classificação. Os relatórios de classificação incluem as seguintes métricas:
  - **precision**: acurácia, que é proporção de acertos de uma determinada classe;
  - **recall**: sensibilidade, que indica a habilidade do modelo de encontrar todas as amostras "Y"ou "N"; f1-score: média harmônica entre as métricas precision e recall;
  - **support**: quantidade total de amostras pertencentes a cada classe.

Além disso, também são reportados valores de médias específicas, que são: macro avg: média por rótulo não-ponderada; e weighted avg: média por rótulo ponderada pelo "support".

Capítulo 3

Técnicas de Treinamento, Desenvolvimento e Validação

Apesar de problemas com exfiltração de dados não serem novos, ano após anos as técnicas utilizadas para roubo de informações se aperfeiçoam. Portanto, qualquer técnica aplicada ao seu combate deve ser extremamente eficaz, e trabalhar bem com baixa quantidade de dados disponíveis, ou grande capacidade de adaptação. Este capítulo irá apresentar as técnicas aplicadas na confecção deste trabalho. E está organizado da seguinte forma, a seção 3.1 abordará o uso do Meta-Learning (MetL), Algoritmo Genético (AG) 3.1.1 E *autoML* 3.1.2. A seção 3.2 discorre sobre o Few-Shot Learning (FSL). A seção 3.3 apresenta o uso e cálculo da Entropia. E a seção 3.4 aborda as métricas de validações aplicadas.

3.1 Meta-Learning (MetL)

O Processo de Meta-Learning (MetL) é o processo de aprender a aprender. Nesse sentido, a ciência do MetL foca em utilizar o conhecimento sobre tarefas anteriores, para aprender a como realizar novas tarefas. Dessa forma, digamos que exista um banco de dados com diversas tarefas resolvidas por inteligências artificiais no contexto da cybersegurança e da exfiltração de dados. Porém, neste momento, temos a tarefa de treinar uma nova inteligência artificial para detectar um novo tipo de exfiltração de dados detectado por especialistas. O MetL nos ajudará a utilizar todo esse conhecimento, para construir rapidamente e de maneira mais eficiente essa nova inteligência artificial, com base nos conhecimentos anteriores.

Dessa forma, o MetL se encarrega de encontrar semelhanças entre determinadas tasks, verificar como o processo de aprendizado de uma inteligência artificial vem ocorrendo e intervir de formar que este processo acelere e aumente a sua eficácia. Além disso, podem

ser realizadas transferências de conhecimento de uma task para outra task, quando tal tarefa for julgada útil. Por exemplo, um processo de pré-treinamento pode ser considerado uma transferência de conhecimento, pois pode-se utilizar uma parte da rede pré-treinada e aproveitar aquele conhecimento semelhante em outra rede. O MetL indicará quais as melhores maneiras de cumprir essa tarefa, como construir as topologias e qual seria a eficiência dessa topologia.

Em sua grande parte o MetL é composto por algoritmos de aprendizado genético, responsáveis por fazer buscas intensas pelas possibilidades tomando como base o cruzamento de informações e semelhanças entre tasks. Além disso, algoritmos de *autoML* (algoritmos responsáveis por construir arquiteturas de redes neurais) podem ser treinados neste ambiente e realizar tarefas parecidas. Tal como outros algoritmos auxiliares podem utilizar machine learning para aprender a prever a performance de um modelo sem a necessidade de treiná-lo, aprender a prever a complexidade de um determinado problema, dado o histórico, e a calcular o custo benefício de resolvê-lo

Nesse sentido, existem no mercado algumas iniciativas relativas à plataformas para desenvolvimento colaborativo de algoritmos de machine learning. Uma dessas plataformas, contendo funções únicas, é a plataforma de MetL da Panaceia. Escolhemos essa plataforma para fazer o desenvolvimento dos algoritmos, assim, realizamos toda a captura dos meta-dados de treinamento pela plataforma, usamos suas funções de algoritmo genético 3.1.1.

Outras vantagens prometidas para o futuro é o uso do MetL em tarefas multimodais e para transferências de conhecimento, como temos em [32], e cuja será facilitado nesta pesquisa pelo uso de uma plataforma de MetL, mesmo que ela venha a mudar, poderemos manter os meta-dados. Além disso, uma das tarefas que serão citadas na revisão da literatura se referem à construção futura de algoritmos de autoML capazes de gerar novos modelos defensores em um tempo curto, tarefas parecidas também foram realizadas em [33] e serão também facilitadas pelo uso de uma plataforma de MetL, e do MetL em si.

Este projeto fez uso das tecnologias totalmente desenvolvidas pela Panaceia, a plataforma de MetL foi desenvolvida inteiramente no ambiente da Panaceia e não teve relação com o desenvolvimento deste projeto, nem usou recursos oriundos da Universidade de Brasília para sua confecção. A plataforma foi escolhida para ser usada devido às suas tecnologias únicas no mercado e ao fato de os membros da equipe já possuírem facilidade em seu manuseio. o projeto foi construído a partir de uma licença especial fornecida pela Panaceia, permitindo a divulgação científica e o uso das topologias de Inteligência artificial geradas como a Universidade de Brasília e os pesquisadores deste projeto desejarem.

3.1.1 Algoritmo genético

Algoritmos genéticos são técnicas de buscas, derivadas dos algoritmos evolutivos AE. São algoritmos simples, e extremamente eficientes para busca em espaços complexos. Que simulam o processo da seleção natural dos melhores indivíduos. Define por indivíduos possíveis soluções ao problema. E população o conjunto de indivíduos considerados. Em [34] os autores definem o algoritmo genético, AG, como um método de busca com heurística adaptativa baseado em genética de populações, introduzido por John Holland no início dos anos 90. Um algoritmo de busca probabilística baseado na seleção natural. O algoritmo é iniciado com uma população inicial. A cada ciclo, chamado de geração, cada indivíduo, ou solução, é avaliado de acordo com seu *fitness value*, ou desempenho para o problema proposto. E ao final da geração são selecionados probabilisticamente de acordo com sua pontuação. Pode ocorrer também a combinação de indivíduos para produção de novas soluções. A cada nova geração, novos indivíduos melhores são gerados. O algoritmo para ao encontrar uma solução ótima, ou, após um certo número de gerações. O algoritmo 2 define um pseudo-código de um algoritmo genético.

Algorithm 2 Pseudocode for genetic algorithm

```
The input and output shapes are defined and receive an Input layer and an output
Dense layer without activation function. A hidden layer of one neuron is defined.
The number of generations to be found is defined.
The loss function to be used on the hidden layers is defined.
The max number of neurons on one dense layer is defined.
The patience for a callback is defined.
The data to train is defined.
while the number of generations don't is reached do
    generate a san with no alterations from actual_generation.
    In the san randomly dense hidden layers can be added or removed with 50% of
probabilite.
    In the san randomly the number of neurons of a hidden layer is changed.
    The san network is trained with callback with patience like chosed.
    if San_loss_function < Father_loss_function then
        actual_generation = San
        generation_number = generation_number + 1
        send_meta_data()
    else
        actual_generation is not actualized.
    end if
end while
```

Apesar de se tratar de um algoritmo de busca, AG também são aplicados em técnicas de IA, [35]. Esses algoritmos são extremamente eficientes na busca pela melhor modelagem de um modelo de IA. Neste caso, o indivíduos são modelos de inteligência artificial,

gerados de forma aleatória. Ao final de cada geração a pontuação de cada membro da população é calculada pela proximidade de sua solução ao problema, ou seja, sua acurácia. Os melhores indivíduos possuem alta probabilidade de serem selecionados. A maioria dos métodos de seleção são projetados para que uma pequena porção de indivíduos com baixa pontuação sejam selecionados, mantendo a diversidade da população. Alguns dos modelos selecionados são combinados para geração de novos indivíduos, processo conhecido como *crossover*, ou combinação/mutação. Indivíduos gerados a partir desse processo compartilham inúmeras características de seus "pais". Técnicas de AG são aplicadas na área da robótica [36], jogos [37], inteligência artificial [35, 38, 39], processamento de imagens [40], etc.

3.1.2 AutoML

Muito autores não consideram o *autoML* como um campo do Meta-Learning (MetL), no entanto, segundo as definições do MetL decidimos considerar um campo do MetL, conforme diversos outros autores [22]. O *autoML* munido de MetL neste trabalho está sendo preparado a partir da coleta dos meta-dados de arquiteturas de modelos e suas performances sobre as tarefas a que são submetidas. Nesse sentido, gerar um algoritmo de *autoML* consiste em gerar um modelo de inteligência artificial capaz de receber como parâmetros os dados específicos de uma determinada task (ou tarefa) e construir com precisão uma arquitetura de inteligência artificial capaz de executar esta tarefa.

Neste trabalho, o nosso objetivo será construir as pré-condições necessárias para a criação de um *autoML* capaz de construir arquiteturas de inteligência artificial para novos tipos de ataques de exfiltração de dados. Tal tarefa, se alcançada no futuro, permitirá àqueles responsáveis pela proteção de dados uma adaptação rápida e contínua, ainda mais se utilizando plataformas de desenvolvimento de inteligências artificiais como a plataforma de MetL da Panaceia, a qual foi selecionada para a realização deste trabalho, devido às suas características únicas relativas à captura dos meta-dados desejados.

3.2 Few-Shot Learning (FSL)

O Aprendizagem Profunda (AP) é uma das áreas de pesquisa mais populares na atualidade. E foi implementado e adaptado a todas as áreas de domínio da ciência moderna, provando ser extremamente efetivo. Contudo soluções convencionais de modelos de AP necessitam de centenas de milhares de dados classificados, e um alto número de iterações no *dataset* para treinamento [41, 42, 43]. Portanto, tornou-se necessário a criação de modelos capazes de simular a capacidade humana de aprendizagem. Diferentes das técnicas convencionais de AP humanos são extremamente eficientes na identificação de objetos

com um número baixo de exemplos. Os autores de [44, 45] propõem um novo paradigma de aprendizagem a partir de bases de conhecimento com número limitado de exemplos, chamado de One-Shot Learning (OSL), ou Few-Shot Learning (FSL).

Em [20] os autores descrevem FSL como o processo de treinamento de um modelo de AM com pouquíssimos dados de treinamento. Ele é baseado na capacidade humana de generalizar novos conhecimentos com base em exemplos previamente memorizados. Comumente, algoritmos de IA necessitam de bases de conhecimento com centenas de milhares de dados. Em [45] os autores expõem que o algoritmo de FSL são adequados em casos onde o baixo número de exemplos são relativos à problemas éticos, de segurança ou privacidade.

Os autores de [41] argumentam que modelos de FSL focam no aprendizado de novas categorias com baixo número de exemplos, e geralmente são combinados com técnicas de MetL, 3.1. Similarmente ao experimento proposto por este trabalho, que utiliza técnicas de FSL e MetL para captura e estudo dos *hyper-parâmetros* dos modelos propostos.

3.3 Entropia

A entropia é uma medida de quantidade de informação no desenvolvimento da teoria da informação de *Claude Shannon*, [46]. *Claude Shannon*, engenheiro e matemático, é considerado pai da teoria da informação. Introduziu no ano de 1948, no trabalho *A Mathematical Theory of Communication*, conceitos primordiais que originaram a teoria da informação. Entre os conceitos propostos está a **Entropia**. Nesse trabalho, *Shannon* desenvolve um modelo teórico para o problema da comunicação envolvendo a transmissão de mensagens digitais. Para ele, é fundamental a transmissão de uma mensagem remotamente, com um grau de fidelidade desejado, independentemente de seu significado. Segundo ele, o processo de geração de informação é um processo não-determinístico, pois, não é possível determinar a priori o canal onde será transmitido. *Shannon* definiu o cálculo da Entropia na equação 3.1, sendo H a entropia, x um evento dentro do conjunto de possibilidades, N o número de elementos, e $P(x_i)$ a probabilidade de ocorrência de i. Na fórmula é calculado o logaritmo de uma probabilidade, que por definição é um número entre 0 e 1, portanto, o resultado é um número negativo, e portanto é colocado um sinal negativo no início da equação, que serve para tornar o resultado um número positivo.

$$H(x) = -\Sigma_{i=0}^{N-1} P(x_i) \log_2 P(x_i) \tag{3.1}$$

Neste trabalho, o cálculo da entropia foi utilizado para priorização de informações no modelo de redes neurais densas, 2.3. Esta demonstra ser uma abordagem computacionalmente leve. Assim sendo, é adicionado um modo de pré-processamento aos dados de

entrada do modelo de RD, com o cálculo de entropia de IP de origem e destino. Com isso, é possível observar que com o crescimento do número de pacotes com mesmo IP de origem/destino, o valor do cálculo de entropia aumentará proporcionalmente.

3.4 Métricas de validação

Para validar a performance dos resultados obtidos, este trabalho adota as seguintes métricas nas análises temáticas e estudos de casos

- **Confusion Matrix**: por meio da matriz de confusão, calcula-se a quantidade de falsos positivos e falsos negativos, bem como de verdadeiros positivos e verdadeiros negativos, propiciando calcular acurácia e sensibilidade;
- **Classification Report**: relatório com as principais métricas de classificação. Os relatórios de classificação incluem as seguintes métricas:
  - **precision**: acurácia, que é proporção de acertos de uma determinada classe;
  - **recall**: sensibilidade, que indica a habilidade do modelo de encontrar todas as amostras "Y"ou "N"; f1-score: média harmônica entre as métricas precision e recall;
  - **support**: quantidade total de amostras pertencentes a cada classe.

Além disso, também são reportados valores de médias específicas, que são: macro avg: média por rótulo não-ponderada; e weighted avg: média por rótulo ponderada pelo "support".

Capítulo 4

Metodologia

O Ramo da cibersegurança é extremamente amplo, e está em expansão. Portanto, para cumprir os objetivos específicos deste trabalho foi preciso construir uma forte estratégia de pesquisa. Este capítulo discorre sobre a metodologia e estratégia de pesquisa para construção do arcabouço das soluções propostas. E está organizado da seguinte forma: A seção 4.1 apresenta a pesquisa realizada para confecção da revisão da literatura. A seção 4.2 expõe a forma de confecção do trabalho. E a seção 4.3 apresenta o banco de dados de referência utilizado para validação das técnicas abordadas. A última sessão discorre sobre os motivos técnicos para a escolha de cada modelo testado.

4.1 Pesquisa

Este estudo visa analisar e propor, do ponto de vista de Inteligência Artificial, contramedidas para detectar ataques de exfiltração de dados através da observação do fluxo de dados em trânsito de uma infraestrutura para detectar e identificar Advanced Persistent Threat (APT). A análise também será feita tomando em consideração os sistemas de Data Lacking Protection System (DLPS), uma adição ao trabalho de [1], que não levou em conta esse tipo de sistema no contexto da defesa cibernética.

Os problemas de APT e exfiltração de dados são um dos maiores problemas no contexto da cibersegurança. Hoje em dia, estes ataques podem ser evitados ou ter suas consequências reduzidas a partir da detecção da exfiltração diretamente no fluxo de dados em trânsito. [47] resolve o problema de exfiltração de dados por Domain Name System (DNS) utilizando técnicas de inteligência artificial. Por outro lado, investigar o fluxo de dados em trânsito pelo protocolo TCP/IP também abarca a exfiltração de dados, via DNS, entre várias outras situações, uma vez que as conexões passam pelo protocolo TCP/IP.

[48] propõe uma solução utilizando ML para a detecção de exfiltração de dados em trânsito, via protocolo TCP/IP, no entanto, este trabalho não leva em consideração os

novos problemas trazidos pelos sistemas em nuvem, onde os dados são largamente pulverizados o que acrescenta novos níveis de complexidade. A nossa investigação buscará resolver esse problema, levando em conta a pulverização dos dados através da observação dos meta-dados dos pacotes de interesse, ou mesmo da busca de padrões ainda não conhecidos por meio de técnicas de mineração de dados, para a posterior construção de algoritmos mais robustos de inteligência artificial, com maior capacidade de auto-adaptação e aprendizado como é o caso do Few-Shot Learning (FSL) e do Meta-Learning (MetL).

Para confecção da revisão da literatura deste trabalho foram utilizados o sistema de "Alerts" do Google Scholar, *Google Scholar* para busca ativa, *Mendeley* para seleção de artigos e algoritmos automatizados para busca de artigos a partir de palavras-chave. Portanto, um conjunto de palavras-chaves foram selecionadas com base no problema, temática e questões da pesquisa. O núcleo do conjunto é composto principalmente por: exfiltração de dados, detecção, prevenção, Aprendizagem de Máquina (AM) e MetL. Em [1] os autores mostram o aumento da eficiência na busca por referências ao empregarem sinônimos de suas palavras-chaves para busca. Por fim, o conjunto de palavras foi combinado por meio de operadores lógicos *AND* e *OR* para confecção de uma *String* de pesquisa completa, figura 1.1. E, foi utilizado as bases de dados *IEEE* e *ACM*, para busca dos artigos.. Foram selecionados artigos, os quais foram classificados de acordo com os seguintes critérios: Latência; Tempo de convergência; Capacidade de convergência não equilibrada.

4.2 Confecção

Neste trabalho foi utilizado o *Wireahark* para coleta de dados de tráfego de transferência de arquivos entre a nuvem, AWS S3. O pacote de dados é analisado a partir da metodologia Cross-industry standard process for data mining (CRISPM) com as linguagens de programação R e Python. A partir desta análise são selecionadas as variáveis mais adequadas para aplicação de Aprendizagem Profunda (AP). Todos os algoritmos foram implementados com o pacote Keras e Tensor Flow, exceto os algoritmos de Aprendizagem Diferencial (AD), que serão implementados pela própria equipe. O objetivo principal da análise será gerar uma inteligência artificial capaz de identificar a diferença entre um pacote malicioso e não malicioso em tempo real.

Uma das abordagens de treinamento foi proposto utilizando um tempo de pré-treinamento com Data Augmentation (DA) por meio de redes generativas para aumento do número de exemplos de exfiltração. Os algoritmos foram comparados passando pela fase de pré-treinamento e não passando pela fase de pré-treinamento.

Todo o trabalho será acompanhado por técnicas de MetL inicialmente focadas em metadados de treinamento, e futuramente poderá evoluir para técnicas de AutoML, apren-

dizado genético, entre outros. Todos os metadados de treinamento, topologia e tarefas relacionadas à detecção de exfiltração de dados com dados em trânsito serão registrados em um banco de dados, que será utilizado para entender o processo de aprendizado dos algoritmos, a sua performance e selecionar qual o mais adequado. Nesse contexto, serão testadas e comparadas diferentes técnicas de ML, como: FSL 3.2, Once Learning (OL), redes densas 2.3, computação por reservatórios e redes diferenciais.

Uma vez que a análise dos metadados gere o meta-conhecimento sobre os modelos, o modelo mais adequado para a nossa tarefa será selecionado e passará por uma fase de *tuning* também utilizando técnicas de MetL, com algoritmos sendo utilizados para *tuning* a velocidade de convergência em variações da tarefa especificada. Um subproduto deste resultado será um meta-modelo capaz de gerar e/ou adaptar um algoritmo capaz de lidar com novos casos de exfiltração de dados nunca antes vistos.

A validação dos resultados ocorrerá com parâmetros de acurácia tradicionais, as curvas de aprendizado e a divisão dos dados em conjunto de suporte, conjunto de treinamento, conjunto de testes e conjunto de validação. Um trabalho futuro será expandir esse algoritmo para ser robusto em casos de corrupção de dados por um adversário, em um contexto de ataque de exfiltração de dados com uma possível tentativa de confusão dos algoritmos com a própria maneira em que os dados são postos em trânsito por um adversário, o próprio MetL possui técnicas para cumprir essa tarefa. A figura 4.1 apresenta o ciclo de vida da pesquisa proposta pelo projeto, contemplando a análise do tráfego com a AWS. Nela é possível perceber que o trabalho concentra-se nos dados em voo, ou seja, no trafêgo de dados de comunicação entre a máquina do usuário, possivelmente infectada, e os ambientes da AWS. O desafio todo está concentrada em conseguir distinguir pacotes TCP/IP enviados de forma voluntária pelo usuário responsável pela máquina, e pacotes enviados de forma involutória. A figura também apresenta o banco de dados de Meta-Learning (MetL), utilizado para guardar os meta-dados e hyper-parametros dos modelos em treinamento, à medida que os mesmos são construídos. A seção 6.1 abordará sobre o uso e vantagens do MetL.

Por fim, este trabalho visa comparar modelos adequados para utilização, em tempo real, para o problema abordado. E, também, validar os modelos abordados por meio de uma base comportamental de atividades humanas, Reconhecimento de Atividade Humana (RAH). Ao final deste trabalho, será possível compreender as vantagens e desvantagens de cada um dos modelos, bem como, as particularidades do problema e sua necessidade por novas soluções.

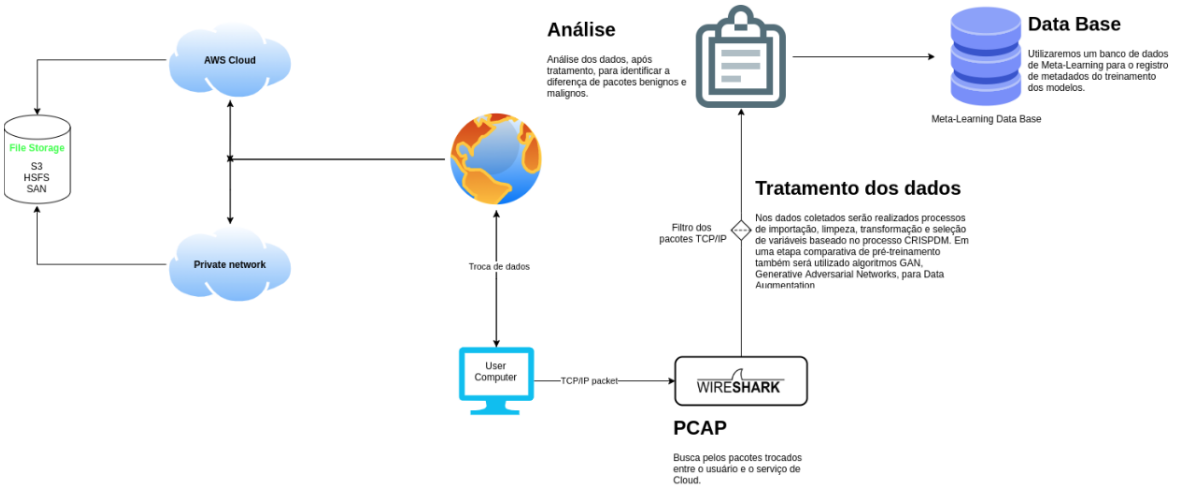

Figura 4.1: Ciclo de vida da análise do tráfego de dados

4.3 Bancos de dados de referência

Tendo em vista que o grande objetivo desse trabalho é a construção do ambiente de treinamento, banco de dados e validação de algoritmos para realizar a detecção da exfiltração de dados, e como os dados de exfiltração são um banco de dados próprio e desconhecido da comunidade, utilizamos o *dataset* UCI [49], no domínio de Reconhecimento de Atividade Humana (RAH) para validar todos os algoritmos. Todo o *dataset* [50] é composto por uma sequências de sinais de giroscópios e sensores de *smartphones*, aonde cada sequência foi capturada enquanto o usuário do *smartphone* estava realizando uma entre seis atividades, como correr, estar parado entre algumas outras.

O Giroscópio é um sensor, fig. 4.2, popularmente presente em *smartphones*, que detecta a velocidade angular do aparelho. Portanto, ele consegue detectar os giros realizados pelo *smartphone* e torno do seu próprio eixo. Essa tecnologia, aliada ao uso de AM, é aplicada a detecção de quedas de idosos.

Nos modelos de detecção de atividades humanas, RAH, é muito comum encontrar atividades fora da distribuição, Dados Fora de Distribuição (DFD), invisíveis em relação as atividades conhecidas e utilizadas para o treinamento do modelos. Nesse sentido, esse *dataset* é extremamente semelhante ao fluxo de pacotes, pois indica uma sequência de estados provocadas por ações humanas. E as atividades espúrias, ou de vazamento de dados, é um atividade fora da distribuição, DFD. Os autores de [50] propõem que é preciso a introdução da incerteza nos modelos, para que estes sejam capazes de detectar atividades fora de sua zona de treinamento. Para isso, eles usam o cálculo de entropia preditiva.

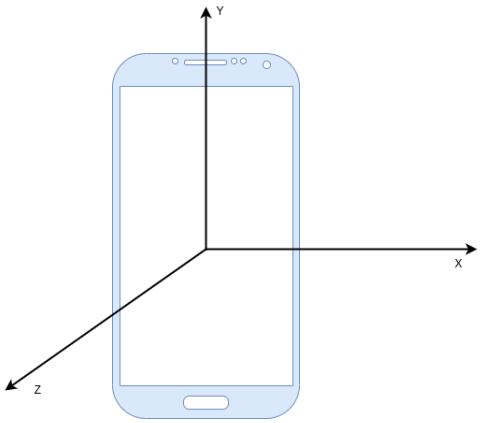

Figura 4.2: Giróscopio em um Smartphone

Portanto, este *dataset* foi escolhido como base para demonstrar a eficiência dos algoritmos propostos em um banco de dados conhecido. Os validando, para poder em seguida, verificar se é factível resolver o problema de exfiltração com os mesmos algoritmos. Nesse sentido, o objetivo final será validar os algoritmos que resolvem o dataset UCI [49, 50], e todo resultado sob os dados de exfiltração deverão ser vistos como embrionários, pois o objetivo é construir as bases para a pesquisa do TransLab acerca da exfiltração de dados. Culminando no cumprimento da tarefa após concluir o projeto de iniciação científica.

4.4 Argumentos para a seleção dos modelos

Como visto durante a sessão 3 abordamos especificamente alguns tipos de modelos de machine learning. Nesse sentido, estes modelos foram escolhidos dentre muitos outros por diversas razões. Existem diversos modelos de machine leaning capazes de lidar com classificação de séries temporais. No entanto, é bem conhecido a maioria dos modelos não inseridos no contexto do deep learning, usam muitas vezes de técnicas heurísticas para resolver os problemas, o que acaba sendo inviável para o nosso contexto, uma vez que o nosso domínio de dados é extremamente complexo. Dessa forma, o deep learning devido à sua grande habilidade para aproximar funções, e alta flexibilidade, é a melhor escolha para o nosso caso de uso.

Dentre as variadas técnicas de deep learning, existem diversas opções de algoritmos capazes de lidar com séries temporais. Dentre elas, umas das mais conhecidas são as redes RR e RMLP que foram utilizadas durante décadas [51] para resolver problemas de séries temporais e ainda o são. No entanto, estes modelos possuem o que é chamado de memória de curto prazo, possuindo uma limitação no que concerne a janela de dados que são capazes de observar. Além disso, suas previsão precisam de O(n) temporalmente para serem executadas, dificultando a sua utilização em tempo real.

Dessa forma, o transformer, por poder executar em O(1) uma série inteira, apresentou vantagens consideráveis com relação às opções anteriores. Além disso, o transformer possui um mecanismo de atenção e uma janela de dados infinita, como explicado na sessão 2 o que permite ao transformer a realização de tarefas extremamente complexas, com resultados abertos impressionantes disponíveis para toda a comunidade.

Já os modelos robustos são outros modelos que possuem uma grande evolução nos últimos tempos com características muito interessantes: Capacidade de adaptação rápida em tempo real, e garantia de não explosão dos pesos para entradas limitadas. Essas duas características combinadas com a capacidade de analisar séries temporais podem ser muito importantes para a criação de um sistema de DLPS verdadeiramente eficiente, e é por esse motivo que essas redes também foram inseridas. Além disso, apesar de exigirem O(n) para serem executadas, na maioria das vezes também são pequenas.

Por último, o modelo de redes densas usando entropia apresentou resultados comprovados em outros trabalhos semelhantes no domínio da cybersegurança, como comentado na sessão 2. Dessa forma, esse modelo foi escolhido como uma base de comparação, devido à dois fatores: Sua modernidade, estando no estado da arte em alguns casos de uso, e seu comportamento bem conhecido pela comunidade científica.

Capítulo 5

Caracterização do Ataque e Defesa Cibernéticos

Os autores de [52] argumentam que o crescimento desenfreado da internet levou a um crescimento expressivo de ataques cibernéticos. Os autores de [1] relataram que ataques de exfiltração de dados afeta todos os tipos de agentes, indivíduos, empresas públicas e privadas, e até mesmo organizações governamentais, custando em média, até o ano de 2021, pelo menos US $3,86 milhões por incidente. Este capítulo, portanto, aborda a contextualização de ataques cibernéticos, especificamente ataques de Advanced Persistent Threat (APT). E esta dividido da seguinte forma: A seção 5.1 aborda ataques de APT. E a seção 5.2 apresenta a relação entre ataques cibernéticos e a base de dados de referência, 4.3, utilizada para validação do trabalho. A última sessão discorre sobre a construção e uso de uma banco de dados no domínio do problema construído pela própria equipe.

5.1 Contexto da ameaça persistente avançada

Em um ataque de APT o atacante se configura como uma ameaça complexa e avançada. Nesse contexto, será feito de tudo para vigiar e capturar os dados do alvo da forma menos perceptível possível. Assim, o atacante toma posse de diversas credenciais do sistema alvo, e se espalha por toda a infraestrutura possível, alcançando cada vez um nível maior de conhecimento sobre o ambiente violado.

Destarte, após ser capaz de obter as credenciais, as exfiltrações de dados de interesse começam, e tentam ser o mais imperceptível possível. Neste trabalho identificamos um tipo de exfiltração de dados que ainda não foi documentado pela comunidade científica, ao menos segundo a nossa revisão bibliográfica. O ataque está focado no ambiente da nuvem, aonde é mais difícil detectar um ataque devido à pulverização dos dados entre

várias estruturas diferentes interconectadas, a exemplo do Hadoop Distributed File System (HDFS) [53].

HDFS é um dos *frameworks* mais utilizados nos ambientes em nuvem da atualidade. Seu fácil acomplamento, escalabilidade e manuseio são características extremamente atrativas. Portanto, naturalmente, é de suma importância que o mesmo possua um forte mecanismo de segurança. Porém, não é exatamente o que acontece. Diversos artigos exploram a falta de segurança inerente ao sistema HDFS, por exemplo [13, 14, 15] entre outros. Em [54] os autores abordam soluções para proteção de dados armazenados no ambiente HDFS. Esses artigos são apenas alguns de milhares de artigos focados na segurança de dados armazenados em ambiente em nuvem, e abordem uma parte do problema deste trabalho, e também, comprovam a extrema necessidade de soluções de segurança para ambientes digitalizados.

A caracterização do ataque parte do pressuposto de que existe um arquivo de extrema relevância à empresa e ao atacante; que a máquina e todas as credenciais da vítima já foram subtraídas, pois considera-se a ação de uma Advanced Persistent Threat (APT); que a máquina da vítima está configurada com um sistema operacional windows; e que o arquivo será arquivado em uma solução em nuvem, como é o caso da S3 AWS - este caso principalmente, pois sabe-se que o arquivo será distribuído em vários servidores, conforme protocolo de armazenamento do HDFS. Definidos esses pressupostos, o ataque ocorrerá da seguinte forma: um e-mail é enviado para a máquina (já comprometida!) da vítima, a vítima em um dado momento acessa o e-mail e clica em um link que executará um comando de linha no CMD (sistema operacional windows) que enviará o arquivo-alvo para a conta da S3 AWS da vítima (também sob domínio do atacante). Nesse caso, o arquivo saiu do domínio da vítima sem o seu consentimento e até agora, sem que a empresa possa realizar qualquer ação que venha a detectar e impedir que o tráfego ocorra e o arquivo seja exfiltrado.

Esse trabalho, portanto, foca na utilização de inteligências artificiais e AM para detectar quando um envio malicioso está sendo realizado para a própria nuvem do ambiente atacado, detectando comportamentos estranhos referentes ao trânsito de dados entre infraestrutura local e nuvem. Nesse sentido, coletamos o trânsito de pacote em uma infraestrutura defendida, e treinamos os algoritmos para detectar os comportamentos suspeitos que podem indicar uma possível exfiltração de dados do tipo descrito.

O algoritmo deverá, quando finalizado, funcionar em tempo real e vigiar o fluxo de pacotes em uma determinada infraestrutura, impedindo uma exfiltração antes que a mesma ocorra. Assim, os próprios algoritmos possuem limitações técnicas a serem consideradas, devendo conseguir ser executados com um tempo de resposta adequado para a tarefa. A partir de uma detecção de exfiltração, o fluxo de pacotes deve ser interrompido.

5.2 Relação com o dataset UCI RAH

Este trabalho objetiva-se na detecção de atividades humanas ou entidades artificiais. Portanto, é crucial para a detecção de ataques de exfiltração que seja possível detectar atividades humanas normativas em um ambiente natural. Os pacotes que atravessam a infraestrutura e são interessantes para o caso de uso, são em grande parte consequências da atividade humana. Nesse sentido, pode-se considerar o problema como um problema de Reconhecimento de Atividade Humana (RAH), ou detecção de atividade humana.

Pode-se levantar hipóteses não experimentais sobre quais padrões seriam observáveis quando houver um ataque, ou um envio normal de dados pela nuvem. Por exemplo, quando um usuário está usando normalmente o computador, e acessando o e-mail será comum ver pacotes de dados referentes a provedores de Simple Mail Transfer Protocol (SMTP), jogos, vídeos no *Youtube* e inúmeras outras atividades rotineiras de usuários utilizando a *Internet*. No entanto, quando se faz um envio proposital de arquivos para a nuvem é possível ver pacotes relacionados a provedores de serviços de nuvem e uma quantidade menor de pacotes relacionados a outras atividades no computador, pois o usuário estará concentrado em enviar o próprio pacote. Nesse caso, um potencial algoritmo capaz de detectar uma exfiltração de dados poderia rastrear tais atividades por meio de seus pacotes, a fim de identificar o momento em que temos o envio do pacote, mas outras atividades rotineiras ainda continuam acontecendo, o que pode indicar um envio não proposital.

Assim, como já foi citado, a observação da atividade humana é um fator crucial para os algoritmos obterem sucesso. No entanto, simultaneamente, o banco de dados relacionado a envios de pacotes, e exfiltração de dados, é novo e carece de validação científica. É conhecido que muitas vezes em uma pesquisa científica os resultados podem não ser bons devido ao próprio banco de dados que se tem a disposição. Nesse sentido, esse trabalho tem o grande objetivo de criar a infraestrutura básica para o TransLab resolver os problemas de APT, mas, ao mesmo tempo, também valida os algoritmos e métodos propostos em um banco de dados já conhecido pela comunidade científica.

É de suma importância verificar se as modelagens construídas realmente conseguem identificar atividades humanas, e consagrá-las como boas soluções, com uma base de comparação científica rica. Portanto, os testes das modelagens sobre o UCI RAH são resultados extremamente valiosos para o TransLab. Em seguida, um teste e seleção inicial desses algoritmos no ambiente de exfiltração de dados, deverá trazer grandes resultados para a pesquisa, levando ao descobrimento de uma modelagem mais eficiente para resolver o problema, que deverá continuar a ser aprimorada pelos próximos anos.

5.3 Testes de envio de dados pelo terminal ou navegador

Até o momento da confecção deste trabalho não foi possível encontrar um banco de dados com tema de exfiltração de dados com ênfase em ataques de APT. Portanto, para desenvolvimento inicial deste projeto foi utilizado como validação dos modelos a identificação de envio de dados para AWS via terminal ou navegador.

Construir modelos capazes de responder a essa pergunta pode trazer resultados positivos, uma vez que, em um ambiente, pode-se definir por padrão o envio de dados diretamente pelo navegador. Como o ataque se dá pelo terminal, identificar nesta rede um envio pelo terminal indicaria com altas chances um verdadeiro ataque ocorrendo. Além disso, essa tarefa é muito semelhante à detecção da própria exfiltração de dados, podendo ser utilizada de maneira adequada para modelar exatamente os mesmo modelos que seriam utilizados na exfiltração.

Foi recolhido um total de 12 séries, com 6 exemplos de cada tipo. No entanto, como exposto na sessão de resultados 7, essa quantidade ainda é insuficiente para validar os dados. Como é verdadeiramente complicado construir esse banco de dados, a continuação da construção deste banco de dados ficará para trabalhos futuros.

Capítulo 6

Implementação

Este trabalho possui os seguintes objetivos:

- Estudo do estado da arte de Aprendizagem de Máquina (AM) (Aprendizagem de máquina) no combate a exfiltração de dados;
- Estudo de modelos de AM aplicadas a cibersegurança;
- Propor modelos eficazes para o problema;
- Validar os modelos em um banco de dados conhecido e através das técnicas de Meta-Learning (MetL);
- Fazer testes inicias com banco de dados mínimo de caso de exfiltração de dados mapeado.

Os capítulos anteriores apresentaram o trabalho realizado a cerca do estado da arta de AM, com ênfase no combate a exfiltração de dados. Também foi apresentado o estudo dos modelos adequados, e suas aplicações no ramo da cibersegurança.

Portanto, este capítulo irá apresentar os modelos propostos para o problema de cibersegurança, a seção 6.1 discorre sobre o uso e vantagens do Meta-Learning (MetL). A seção 6.2 apresenta a arquitetura *TCP/IP*. A seção 6.3 apresenta o trabalho realizado a cerca do RD. A seção 6.4 apresenta o trabalho realizado com *transformers*. E a seção 6.5 discorre sobre o uso das redes robustas.

6.1 Uso do Meta-Learning (MetL)

A ferramenta de MetL utilizada permite a captura dos meta-dados dos hyperparametros dos modelos, à medida que eles são construídos. Nesse sentido, quando um modelo é montado devemos definir alguns parâmetros:

- A task que aquele modelo irá resolver.
- O tipo do modelo.
- O contexto em que a task está inserida.
- O formato da entrada do modelo.
- O formato da saída do modelo.
- Os formatos do banco de dados, e a sua localização.
- Os parâmetros de acurácia e métricas de validação utilizados para estes modelos.

Em seguida, a plataforma captura automaticamente os hyperparametros do modelo como:

- O número de camadas.
- O número de neurônios por camada.
- O histórico de treinamento do modelo.
- O desempenho do modelo ao longo do treinamento, através da captura do histórico das funções de perda.
- A topologia final gerada, com os pesos definidos após o treinamento.
- A loss function escolhida para o treinamento do modelo.
- Se existirem, os parâmetros de callback.
- Todos os parâmetros de velocidade do treinamento e constantes inseridas nos modelos pela escolha do usuário.

Esse processo ocorre para cada uma das tarefas mencionadas neste trabalho, tanto na classificação do RAH, como na classificação entre envios de pacotes pelo terminal e/ou navegador. Foram adicionadas pequenas descrições para cada task, conforme encontramos mais abaixo:

Modelo HAR:

- tipo=Transformer cyber. OU robust_classifier_cyber OU dense_genetic_cyber
- parâmetros de acurácia = acurácia, matriz de confusão, revocação.
- descrição = classificação de atividades humanas para validar eficácia de algoritmos.
- macrocontexto = Classificação de ataques cyberneticos - modelo base HAR-UCI.

De forma semelhante, foram adicionadas algumas informações sobre a classificação entre envio pelo terminal, e envio pelo navegador (uma fase anterior à de exfiltração, que será um trabalho futuro).

Estas definições permitirão um desenvolvimento de alta qualidade dos modelos, e além disso, uma geração de um banco de dados próprio para o TransLab para a aplicação em casos cibernéticos. Nesse sentido, consideramos que dominar essa tecnologia, e ter trabalhado no mapeamento de todos os modelos, alcançando com sucesso a captura de todos os meta-dados de treinamento é de extrema importância e foi uma grande conquista desse trabalho.

A análise dos meta-dados permite conhecer de fato a acurácia de cada modelo, os seus detalhes e uma ponderação acerca da complexidade do modelo. Devemos criar diversas análises, como: Ordem de complexidade de execução do modelo, quantidade de parâmetros do modelo e resultados. Assim é possível capturar os custos-benefícios de cada modelo e optar pela melhor escolha. O que deverá ser feito ao longo dos próximos anos, porém já poderemos entregar resultados inicias neste trabalho.

Os meta-dados também servirão no futuro para a construção de autoML's. O processo funciona da seguinte maneira: Uma vez munidos de 3 diferentes tipos de modelos (redes robustas, redes densas e transformers) podemos treinar diferentes redes para cumprir as seguintes tarefas:

- Identificar a rede com o maior custo-benefício para um determinado contexto.
- Gerar a topologia mais adequada para resolver o problema, de acordo com uma caracterização do ambiente e fornecimento do tipo de dados.
- Identificar antes de treinar um novo modelo, a probabilidade deste ser bem-sucedido ou não para um ambiente de defesa específico.

Estas tarefas serão realizadas no futuro, conforme a evolução do banco de meta-dados a partir dos próprios desenvolvimentos de modelos realizados no ambiente. Neste trabalho acompanhamos os meta-dados para analisar os modelos, as métricas aqui inseridas foram calculadas a partir desse banco de dados e testes adicionais, tal como a conclusão sobre as limitações dos modelos, uma vez que analisamos todo o histórico de alterações feitas em cada modelo, inclusive as alterações automatizadas por algoritmos genéticos.

6.2 Arquitetura TCP/IP

O termo TCP/IP é um acrônimo para o termo Transmission Control Protocol/Internet Protocol Suite (TCP/IP), que são um dos mais importantes protocolos utilizados na internet globalmente. O Protocolo IP é a base da estrutura da comunicação da internet,

baseado no paradigma de chaveamento de troca de mensagens, denominadas como pacotes. O protocolo IP, Internet Protocol (IP) é utilizado tanto no modelo TCP/IP, quanto no modelo ISA, Interconexão de sistemas abertos (ISA), [55] . A arquitetura TCP/IP, assim como a arquitetura ISA, é estruturada em camadas. Em [56] os autores explicam que a distribuição dos dados do pacote em camadas facilita sua manutenibilidade e divisão de responsabilidades. Neste trabalho estamos interessados em duas camadas: aplicação e rede.

A primeira camada, importante para o cálculo de entropia, 3.3, é a camada de **rede**. Esta camada realiza efetivamente a comunicação entre máquinas, através do protocolo IP. Ela carrega a identificação de cada máquina, de origem e destino.

A próxima camada utilizado no projeto é a camada de aplicação. Ela transporta efetivamente os dados que são de interesse dos modelos para detecção de exfiltração de dados. Portanto, seu conteúdo é de extrema importância para classificação dos pacotes.

Para confecção deste trabalho foi montado um dicionário de sinônimos para cada variável capturada via *Wireshark* e sua significância para o problema abordado.

6.3 RD genética com entropia, para séries temporais

Para seleção da rede mais adequada foi aplicado ao experimento o uso do Algoritmo genético, AG. Para isso, utilizou-se de 5 gerações, com tamanho de população 500 indivíduos. E paciência de 2.

O mecanismo de paciência limita o número de gerações que não apresentam alterações na média do fitness da população. Ou seja, quando o algoritmo encontra seu ponto de estabilização. Assim, após o número de 2 épocas sem apresentação de uma melhora, a geração é interrompia, e é retornado os indivíduos com melhores *fitness*. O modelo proposto combina o uso do algoritmo de AG com o cálculo de *Entropia*. Para atingir o objetivo, a abordagem proposta seguiu conforme a figura 6.1.

Conforme demonstrado na figura 6.1, após a captura dos pacotes é feito um filtro dos pacotes adequados ao modelo. Logo em seguida, é realizado uma limpeza nos dados fixando-se na camada de aplicação da arquitetura *TCP/IP*, 6.2. Então, é feito o cálculo da entropia com os dados de endereço de origem/destino.

Entropia tem sido utilizado no treinamento de algoritmos de AM, demostrando ser uma abordagem simples, leve computacionalmente e eficaz. Os seguintes trabalhos [57, 58, 59] discorrem sobre a aplicação do cálculo de entropia em modelos de RD para melhorias de sua acurácia. Em condições normais, o trafego de pacotes dentro de uma rede será distribuído por um número elevado de endereços de destino. Em casos de possível exfiltração de dados, um grande número de pacotes será concentrado em um único, ou talvez, em

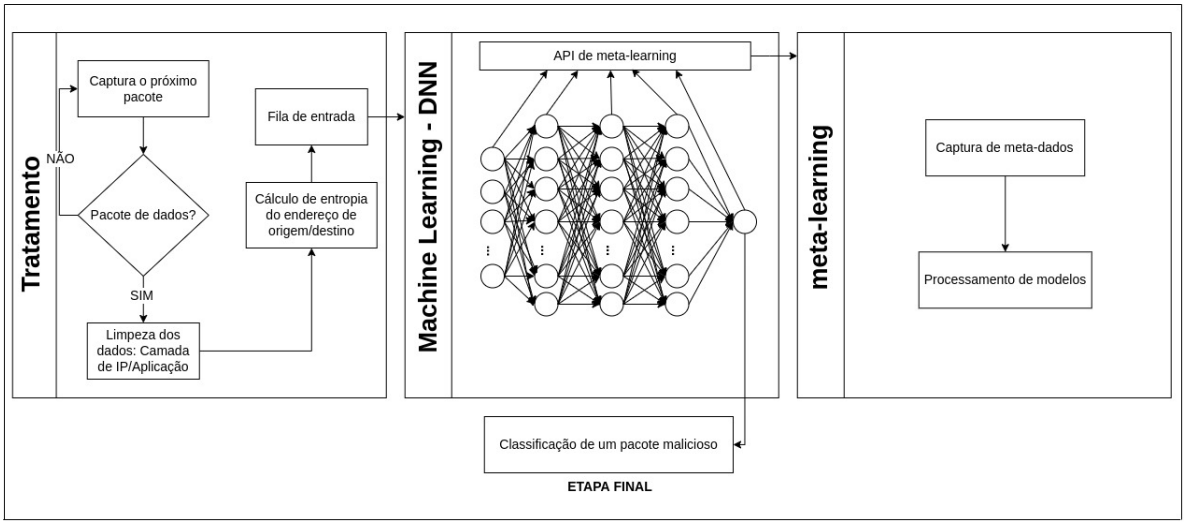

Figura 6.1: Abordagem proposta para detecção de pacotes maliciosos

poucos endereços de destino. O cálculo aplicado a este trabalho é apresentado na equação 3.1.

No caso do *dataset* UCI, [49, 50], cada entrada é uma sequência de 561 *timestamps*, ou 561 registros do giroscópio, fig. 4.2. Assim, cada série foi separada em 561 entradas, e cada entrada foi acoplada ao cálculo de entropia, 6.2.

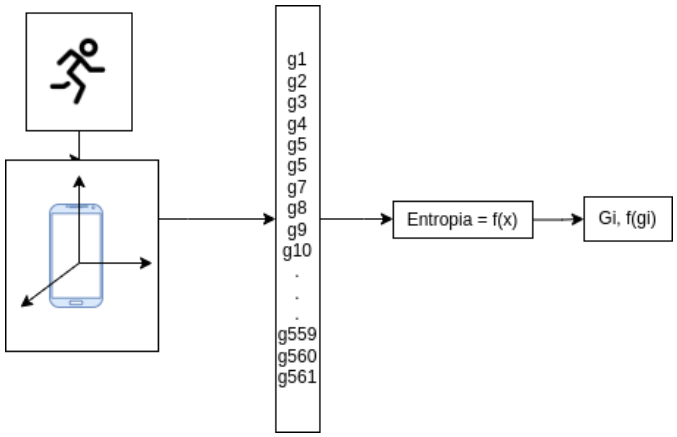

Figura 6.2: Pré-processamento para o modelo UCI

Dessa forma, o modelo mais adequado é uma composição de 4 camadas, com as seguintes características:

- Camanda de entrada: ângulo do giroscópio adicionado o cálculo da entropia
- Camada escondida com 229 neurônios, e função de ativação Rectified Linear Unit (RELU)
- Camada escondida com 128 neurônios, e função de ativação *softmax*

- Camada de saída com 6 neurônios
- *loss Sparse categorical crossentropy*

6.4 Transformer adaptado do text-embbeding

O modelo de transformers disponíveis na tecnologia existente é o mesmo proveniente do artigo Attention is All you need [2]. Portanto, o modelo foi adaptado de um trabalho específico com dados textuais o que limitou de certa forma a organização dos dados de entrada disponíveis. No futuro, isso poderá ser mudado, recomendamos ao provedor uma maior flexibilização da tecnologia de transformers oferecida.

a limitação se dá devido à necessidade do modelo em receber dados no formato de word-embbeding, matrizes quadradas provenientes de uma representação vetorial de diversas palavras. Nesse sentido, para o caso dos pacotes, foram utilizados na entrada uma sequência de pacotes com o mesmo número de atributos selecionados por vez.

No caso do dataset UCI - RAH cada sequência possuía um total de 561 timestamps. Nesse sentido, e sem prejuízo para o resultado final, cada timestamp foi transformada em uma matriz quadrada de 24 por 24. Claramente, o número de parâmetros não é suficiente para preencher alguns espaços da matriz quadradas, os quais foram preenchidos com zeros (assim, não gerando qualquer influência sobre os resultados do modelo).

O que acontece, portanto, no caso do UCI é que o transformer analisa conjuntamente, sem informações temporais, semi-conjuntos de tempos t, e temporalmente cada um dos 24 semi-conjuntos criados pela matriz. O que já dá a noção temporal para o modelo conseguir relacionar padrões entre toda a sequência de maneira eficiente. No caso da análise dos pacotes, cada pacote é analisado também no domínio temporal, e não por conjuntos. No entanto, poderia ser uma técnica pertinente a conjunção de pacotes para mudar o tamanho das sequências, sem prejuízo dos resultados.

Além disso, um problema percebido anteriormente era a falta de capacidade de fazer o modelo treinar para classificar apenas com uma variável binária de saída, ou, por exemplo, 6 variáveis no caso do UCI RAH. Quando tentamos treinar o modelo dessa forma, sempre chegamos na explosão de gradientes, atribuímos a esse fato a evidente desproporção entre a quantidade de parâmetros da rede e a saída gerada, com uma só variável. Tal característica levou os valores do gradiente à grande instabilidade, provocando a explosão.

A solução para este problema foi estender o espaço vetorial da saída. Assim, criamos um espaço vetorial com 561 variáveis, como se trata de um problema de classificação, no caso da exfiltração e detecção de envio pelo navegador ou terminal, com duas classes e no problema do UCI RAH com 6 classes, criamos um vetor aleatório com valores limitados de 0 a 100 para cada uma das classes. Essa mudança na forma de treinamento permitiu

que o gradiente não explodisse, sendo dividido entre todo o espaço vetorial maior da saída e trazendo ótimos resultados.

Nesse sentido, o transformer escolhido como mais eficiente, após alguns testes e análise dos meta-dados gerados foi o transformer com as seguintes características:

- 3 camadas, com 128 neurônios em cada camada de atenção, 8 cabeças
- 128 neurônios nas sub-camadas densas
- tamanho do "vocabulário" de entrada de 24, e tamanho do vocabulário de saída 561.

O transformer foi treinado com o algoritmo heurístico de backpropagation ADAM [60], devido à característica da criação do espaço vetorial maior, e previsões muito restritas, se torna mais fácil fazer a classificação. Portanto, no caso do UCI RAH optamos por utilizar a função de perda do erro quadrático médio, mais abaixo explicaremos como é feita a classificação a partir dos resultados gerados pelo transformer. No caso dos dados de pacotes, usamos a mesma técnica. Primeiro, tentamos utilizar a entropia binária cruzada, no entanto, esta estava levando também aos problemas de explosão do gradiente por exigir apenas uma variável ao final.

Assim, ao usar o erro quadrático médio, obtivemos do transformer um espaço vetorial de 561 valores, que não são exatamente iguais aos vetores definidos como representações de cada classe. Destarte, nós fazemos um cálculo de proximidade, a partir do erro quadrático médio entre os vetores de cada classe e o vetor gerado pelo transformer, aquele que retornar o menor erro quadrático médio, será a classe que o transformer previu. Os resultados foram bastantes satisfatórios a partir dessa técnica. A qual é muito semelhante à técnica utilizada para classificação com redes robustas.

6.5 Redes robustas para a classificação de seis classes

As redes robustas disponíveis são exatamente como aquelas descritas na sessão 2.2. Neste caso, o ferramental oferece uma classe no python responsável por gerenciar todos os regressores, ou seja, redes diferenciais robustas, que irão aproximar cada classe. Nesse sentido, para os problemas de classificação de duas classes como no problema do terminalXnavegador teremos duas redes robustas, uma para cada classe. No caso do UCI RAH teremos seis redes robutas, uma para cada classe.

O processo de treinamento ocorre da seguinte forma: Cada regressor é treinado apenas com as séries de sua classe respectiva, assim aprendendo a fazer a regressão das séries de uma classe, e, teoricamente, somente dela. Assim, no momento da classificação, todos os

regressores são utilizados em uma mesma série, aquele que obtiver o menor erro quadrático médio para a aproximar, será a classe a qual a série pertence.

Por uma limitação da tecnologia, e para acelerar o tempo de treinamento, no caso do dataset UCI HAR passamos e ou 3 tempos t de uma vez para o regressor. Sabemos que isto poderia influenciar os resultados finais, trazendo uma performance um pouco inferior, por uma característica de fator de discretização. Ou seja, estamos saltando a série em espaços temporais maiores. No entanto, podemos fazer um paralelo com modelos recorrentes de memória longa LSTM's [51] podemos argumentar que apenas estamos passando pela série com uma janela de tamanho 3, e provavelmente isso não irá influenciar muito os resultados finais. Essas práticas são corriqueiras em ambientes de processamento de imagens e de texto, e deixam o processamento mais rápido.

Assim, os modelos regressores foram montados com uma camada de entrada de tamanho 3, e uma camada escondida de tamanho 1000, retornando para uma camada de saída de tamanho 3. As leis de adaptações são as mesmas definidas na sessão 2.2. Para o caso dos pacotes, nós temos simplesmente a entrada e a saída com exatamente a quantidade de atributos de cada pacote.

Capítulo 7

Resultados

Para a tarefa de previsão de envios de pacotes pelo terminal ou pelo navegador, iniciamos o treinamento a partir das redes densas. Comparando com os resultados obtidos para o caso do UCI Reconhecimento de Atividade Humana (RAH), chegamos à conclusão de que os dados eram insuficientes para treinar os modelos propostos neste trabalho. A rede densa foi capaz de "decorar" os dados, chegando à acurácias de 100%, fato bem diferente de quando treinadas em um *dataset* mais extenso como o UCI RAH, tal como apresentaremos mais abaixo. Nesse sentido fazemos a recomendação para o TransLab de aumentar os dados, e gerar um banco de dados mais robusto relacionado aos casos de exfiltração antes de treinar esses algoritmos. Inicialmente, o Few-Shot Learning (FSL) poderá ser uma técnica de treinamento viável e adequada, conforme o banco de dados evolui.

Por esta razão, decidimos não treinar os outros modelos neste banco de dados pequeno, uma vez que podemos esperar o mesmo resultado, deixando como trabalho futuro a adequação de técnicas de FSL para os modelos, e a construção de um banco de dados maior. Assim, todos os outros resultados abaixo fazem parte da validação dos modelos para o contexto RAH.

7.1 Distribuição do dado

As classes estão distribuídas da seguinte forma: Existem 7352 exemplos de séries para teste. Cada um dividido em 6 classes: andando, subindo escadas, descendo escadas, sentado, parado e deitado. A distribuição de cada classe nos dados de treinamento é apresentado na tabela 7.1.

Tabela 7.1: Distribuição das classes dos dados de treinamento.

Classe	**Quantidade**
Andando	1226
Subindo escadas	1073
Descendo escadas	986
Sentado	1286
Parado	1374
Deitado	1407

Tabela 7.2: Distribuição das classes dos dados de teste.

Classe	**Quantidade**
Andando	496
Subindo escadas	471
Descendo escadas	420
Sentado	491
Parado	532
Deitado	537

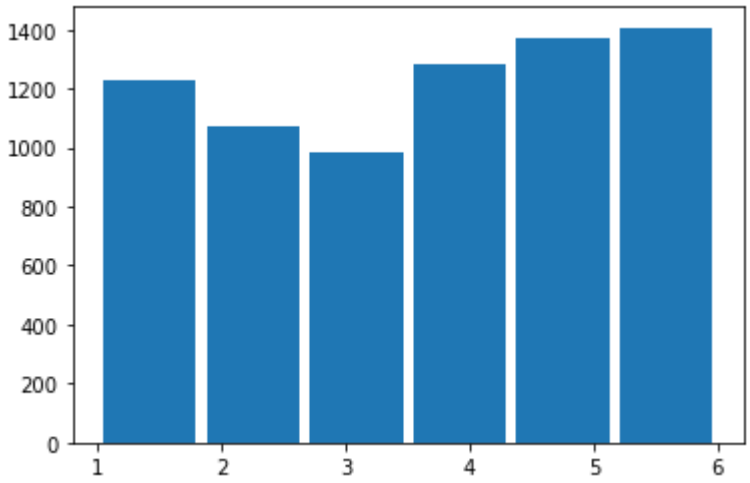

Figura 7.1: Histograma da distribuição de exemplos por classe no dataset, dados de treinamento

A figura 7.1 apresenta o histograma da distribuição de cada classe, identificado de 1 a 6, é possível perceber que os dados estão bem distribuídos. Já para validação dos modelos existem 2947 dados de teste, tabela 7.2.

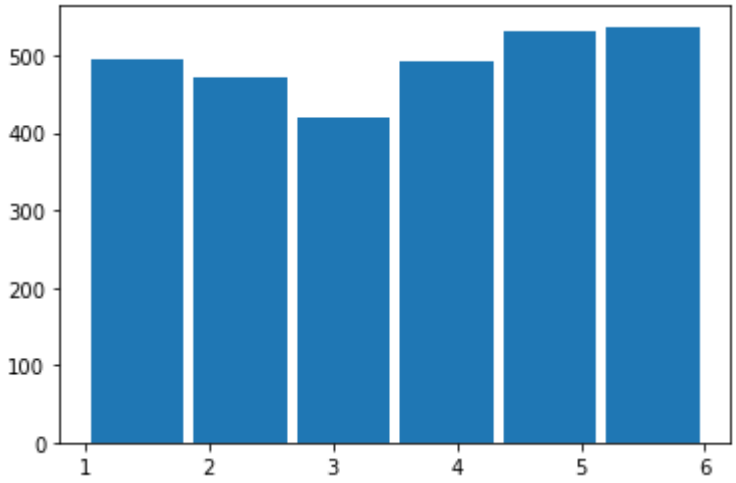

Figura 7.2: Histograma da distribuição de exemplos por classe no dataset, dados de validação

Com relação aos dados de validação, percebe-se que algumas classes possuem menos exemplos do que outras, fig. 7.2. Porém, não há um grande desbalanceamento entre os dados, e dessa forma, podemos utilizá-los como estão, para aproveitar a quantidade de dados.

7.2 O transformer é promissor

O modelo transformer foi treinado pelo total de 300 épocas, pela duração de um dia. Após o treinamento, o modelo foi testado em um total de 30% dos dados, que foram previamente separados para a faze de teste. Dessa forma, o modelo obteve os seguintes resultados:

- Acurácia: 91%
- A tabela 7.3 apresenta a matriz de confusão;
- Recall: 0.91%;
- F1-score: 0.91%.

As métricas obtiveram resultados muito parecidos, sendo suas diferenças em casas decimais, dessa forma deixamos todos como 91%. Esse resultado indica a robustez da medida de acurácia, de forma que não existe algum resultado ruim mascarado pela concisão da medida de acurácia.

Na tabela 7.3 cada linha representa uma classe, e cada coluna representa uma classe, a diagonal representa as classificações corretas, ou seja, quando uma classe foi classificada

Tabela 7.3: Matriz de confusão dos *transformers*.

469	17	10	0	0	0
45	426	0	0	0	0
5	31	384	0	0	0
0	1	0	419	70	1
0	0	0	86	446	0
0	0	0	0	0	537

corretamente pelo modelo. Os outros elementos de cada coluna representam para a classe i, quantas vezes o modelo classificou erroneamente uma série pertencente àquela classe, como sendo de uma das outras 5. Nesse sentido, podemos perceber uma disparidade pequena entre os resultados, e uma alta acurácia. O modelo foi capaz de acertar a maioria dos casos, sendo o maior problema entre as classes 4 e 5 que foram trocadas muitas vezes. Esse fato poderia ser melhorado por meio de um rebalanceamento dessas classes com relação às outras, porém isto não foi realizado neste trabalho.

Como mostrado as classes estão bastante equilibradas, no entanto, há um erro maior entre as classes 4 e 5, que acabam sendo invertidas algumas vezes. É possível suplantar esse resultado com um treinamento maior. Porém, pode-se considerar esse resultado como satisfatório. Na própria literatura, os resultados estão muito próximos dos obtidos, indicando o grande potencial deste modelo, uma vez que ele ainda não foi tunado.

Também podemos confirmar que a distribuição dos dados não afetou a qualidade do modelo, podendo a considerar adequada para o treinamento, uma vez que os casos que deram mais errado, foram aqueles em que havia uma quantidade de exemplos maior.

Recomendamos a aplicação futura de algoritmos genéticos, tal como no caso das DNN's, pois acreditamos ser possível aumentar os resultados preliminares deste modelo que obtivemos neste trabalho.

Podemos concluir que o modelo é adequado para detectar atividades humanas, o validando para ser utilizado futuramente no caso de exfiltração de dados.

7.3 As redes robustas falharam

O modelo de redes robustas foi treinado passando uma vez por cada exemplo, o classificador respectivo, o que levou dois dias. Esse tempo a mais não pode ser considerado como menos eficiente, pois foi usado um código não paralelizado para as redes robustas. Outras tentativas foram feitas, com a camada escondida maior ou menor, e com a variação das constantes e matrizes que determinam a velocidade de convergência do modelo. Após o treinamento, o modelo foi testado em um total de 30% dos dados, que foram previamente separados para a faze de teste.

No entanto, o modelo não conseguiu ultrapassar 18% de acurácia em nenhuma das variações de hyperparametros, indicando uma inadequação do modelo para o problema proposto. Dessa forma, não faz sentido apresentar as outras métricas, uma vez que estas não são relevantes em um contexto em que não se tem nem mesmo uma boa acurácia.

As redes densas possuem a característica de resolver equações diferenciais com muita precisão. Porém sistema que não podem ser modelados por equações diferenciais, muitas vezes possuem características como mudanças abruptas nos dados, ou mesmo, padrões localizados difíceis de serem modelados. Estas são exatamente as características desses dados. Dessa forma, apesar de poderem parecer ser eficientes à uma primeira análise, pode ser que as redes robustas não sejam adequadas para resolver esse problema.

Ainda assim, temos exemplos muito interessantes de aplicações com sucesso de redes robustas em sistemas altamente complexos, como por exemplo, para a classificação de sinais cerebrais [3]. Nesses casos, podemos ressaltar que as redes robustas possuem a capacidade de se atualizarem online para se adaptar à um determinado contexto, e à mudanças localizadas, e foram concebidas para funcionar dessa maneira, apesar de também poderem funcionar de maneira estática e tradicional. Desse modo, é possível dizer que utilizar as redes robustas de outras maneiras, com atualizações online, poderia ser um caminho viável. Porém, se impõe desafios maiores para testar a rede, uma vez que ela sempre será atualizada com qualquer dado de entrada. Ao mesmo tempo, esse tipo de característica é muito importante no ambiente de APT, e por essa razão, acreditamos um importante trabalho futuro seja testar a viabilidades dessas redes em um contexto não estático, aonde se treina e depois avalia estaticamente a rede.

7.4 As redes densas falharam

O modelo de rede densas foi treinado utilizando inicialmente a série completa dos movimentos, RAH. Então, ele tinha como entrada 561 dados lidos do giroscópio, combinado com o cálculo médio da entropia de cada dado. Seja $H(x_i)$ o cálculo de entropia registrado na equação 3.1. O cálculo da entropia média é apresentado na equação 7.1. Onde $S(X)$ representa a série temporal, com N entradas.

$$M(S(X)) = \Sigma_{i=0}^{N-1} H(x_i), x_i \in S(X) \tag{7.1}$$

Inicialmente, era proposto que o cálculo de entropia ajudasse o modelo a convergir de forma rápida. Porém, o uso apenas da média da entropia da série não foi suficiente para que melhorasse sua acurácia, e para que o modelo se adequasse as séries temporais. Assim, foi acrescentado a entrada do modelo os valores de entropia miníma e máxima da série. Porém, ainda sem sucesso. Por fim, o modelo foi adaptado para lidar com valores unitários

dos dados lidos do giroscópio, e não mais com a série completa. Assim, a entrada consistia de apenas um único angula do giroscópio, e do cálculo da entropia daquele angulo. No fim as redes densas não se adequaram bem ao *dataset*. Obtiveram uma acurácia extremamente baixa. Mesmo aliadas ao cálculo da entropia, não obtiveram resultados satisfatórios na base de dados *UCI*. Portanto, sua acurácia manteve-se extremamente baixa. Dessa forma, não faz sentido apresentar as outras métricas, uma vez que estas não são relevantes em um contexto em que não se tem nem mesmo uma boa acurácia. Este tipo de modelo não se adapta bem a entradas com *datasets* temporais, como no caso de pacotes de dados, ou mesmo do RAH.

7.5 Discussão e comparações

Dado os fatos apresentados é possível perceber que dois entre os três modelos não apresentaram os resultados esperados. Mesmo sendo modelos validados anteriormente em outros trabalhos científicos em contextos de dados de séries temporais. No entanto, os dados RAH são mais complexos, pois envolvem padrões localizados e possivelmente envolvendo contextos que dependem de memórias mais longas para funcionar.

Porém, os baixos resultados ainda não invalidam totalmente os modelos de Redes Robustas e Redes densas, pois ainda é possível fazer outras variações e transformações nos dados que estariam fora do escopo deste trabalho. Por exemplo, no caso das redes densas, é possível encontrar maneiras de expandir o espaço vetorial do cálculo de entropia, para este se tornar mais equilibrado com os dados de uma séria inteira. Ou, ainda, fazer uma espécie de sistema de votações, aonde calcula-se a entropia para cada valor da série, e não mais para a séria inteira, e o modelo iria classificar cada *timestamp* da série como pertencente a uma determinada classe. Ao final, a classe da série seria a maior frequência de chutes do modelo para cada *timestamp* da série completa. Essas variações, que sairiam do desenho deste experimento, podem ser consideradas para trabalhos futuros.

O caso do *transformer*, segundo a interpretação deste trabalho, é explicado devido às próprias características do *transformer*: É um modelo capaz de relacionar dados com uma distância grande entre si, possuindo uma janela de análise tão grande quanto o mentor e a máquina permitirem. Além disso, o *transformer* é um modelo totalmente baseado em um mecanismo de atenção, o que significa que o modelo é desenhado para ser capaz de observar subconjuntos dos dados que tenham padrões verdadeiramente interessantes para a sua tarefa.

Assim, como tanto as redes densas quanto as redes robustas não possuem uma janela temporal de análise tão grande, e não são especificamente construídas com modelo de atenção. Por todos esses aspectos é possível concluir que o *dataset* possui características

Tabela 7.4: Comparação entre os modelos.

	Transformers	**Redes densas**	**Redes robustas**
Acurácia	91%	26%	18%

[26] que demandam a capacidade de lidar com padrões localizados entre os dados, e também, possivelmente, relacionar eventos que ocorreram com uma distância temporal esparsa.

Nesse sentido, é possível afirmar que para o contexto de APT o modelo *transformer* se apresenta, até o momento desta pesquisa, como o modelo mais viável, pelos três seguintes pontos:

- Capacidade de lidar com padrões localizados.
- Capacidade de relacionar eventos esparsos.
- Capacidade de realizar uma classificação de uma série inteira em apenas uma previsão $O(1)$ [2].

Como exposto, essas características são exatamente as necessárias para lidar com dados do tipo RAH. E, provavelmente, os dados provenientes do contexto APT possuem as mesmas características, pois se tratam de dados provenientes de atividade humana, ou atividade artificial (um domínio não muito diferente do primeiro em termos de características do dado).

No entanto, devido ao tamanho menor dos modelos de rede robustas e redes densas, e aos resultados iniciais obtidos com um *dataset* pequeno, é intuitivo que ainda não se pode descartar esses dois modelos. Sendo necessário um avanço no sentido de técnicas de Few-Shot Learning (FSL) inicialmente, e algumas variações na modelagem dos modelos. Assim, será possível fazer mais um teste de validação e comparação com os *transformers*. o Meta-Learning (MetL) irá capturar caso a caso, e indicar qual a melhor solução em cada caso específico. Dessa forma, conclui-se que o *transformers* tem o potencial de ser o modelo com a capacidade mais generalista de resolver múltiplos problemas no domínio estudado, porém, em alguns casos, os outros modelos podem acabar sendo opções, por questões de velocidade, custo-benefício e outros fatores citados.

Capítulo 8

Conclusão e Trabalhos Futuros

O Avanço no ramo da Inteligência Artificial (IA) nos últimos anos despertou o interesse, e a necessidade, do seu estudo no ramo da cibersegurança. Não é novidade que a cada ano a atividade de *hackers* se intensificam e causam enormes prejuízos até mesmo as grandes companhias, *Big-techs*. Não somente a essas empresas, mas o vazamento de dados é crítico, até mesmo para grandes nações e seus dados sigilosos de Estado. Portanto, é de suma importância que novas soluções sejam propostas e desenvolvidas. Portanto, o trabalho apresentado ocupa-se no estudo de técnicas e modelos capazes de serem aplicadas no ramo da cibersegurança. Assim, este capítulo apresentará as contribuições realizadas pelas atividades desenvolvidas, seção 8.1. E por fim, apresenta os trabalhos futuros a serem desenvolvidos, seção 8.2.

8.1 Contribuições

Este trabalho apresentou propostas para problemas com cibersegurança, com ênfase em Advanced Persistent Threat (APT). A proposta deste trabalho foi a revisão sistemática da literatura, e do estado da arte de técnicas e métodos de AM com foco em cibersegurança. Além da apresentação de 3 possíveis modelos eficazes para o problema. Também concentrou-se na validação das técnicas apresentadas por meio de um banco de dados conhecido e conceituado. Por fim, este trabalho também aplicou técnicas de Meta-Learning (MetL) para salvar dados e meta-dados de todos os modelos e treinamentos realizados, para que em trabalho futuros seja possível reavaliar e revalidar os modelos trabalhados, assim como estes dados foram utilizados neste trabalho para validar o modelo.

Portanto, o presente trabalho desempenha um papel significativo no estudo de técnicas relacionadas ao combate a exfiltração de dados, mais especificamente as técnicas de APT. Sendo esse um ataque extremamente sofisticado, e de difícil detecção. Conseguimos mapear o estado da arte no campo, e mais especificamente, identificar uma falha relacionada

ao contexto da exfiltração de dados. Além disso, foi possível mapear o estado da arte do uso de técnicas de inteligência artificial para a resolução do problema, garantindo assim que o trabalho estivesse atualizado e relevante para a comunidade científica.

Este trabalho apresentou diversos modelos e técnicas possíveis de serem aplicadas para solução do problema. Também foi apresentado um arcabouço de um sistema de DLPS, que, com o tempo necessário é imensamente factível, através da união desse conceito com sistemas colaborativos de Meta-Learning (MetL), como o utilizado neste trabalho com bastante sucesso.

Para validação das soluções propostas, foi utilizada uma base de dados de atividades humanas, Reconhecimento de Atividade Humana (RAH). É importante salientar que, em um ambiente de exfiltração é possível considerar eventos de vazamento de dados como anomalias, ou eventos não esperados, Dados Fora de Distribuição (DFD). Portanto, a base de dados UCI [49] é amplamente conhecida e adequada para validação das técnicas e modelos apresentados. E dessa forma, conseguimos construir um ambiente de teste válido para os algoritmos modelados e construídos.

Os resultados obtidos comprovam a eficiência e adequação do modelo *transformer* para detecção de atividades humanas, bem como, sua facilidade de uso e de treinamento. Portanto, como próximo passo, é preciso realizar a melhoria dos modelos e a construção de um banco de dados maior de casos de exfiltração de dados. Em seguida, deveremos incluir medidas de complexidade do modelo, para lograr o objetivo de utilizar o modelo em tempo real, testando ativamente os modelos em tempo real.

8.2 Trabalhos futuros

Como trabalho futuro fica a aplicação dos algoritmos desenvolvidos, e validados por meio da base de dados UCI [50], no tráfego de pacotes em uma rede em um sistema de Data Lacking Protection System (DLPS) após o banco de dados ser completado. O trabalho desenvolvido nesta monografia concentrou-se no estudo e caracterização de ataques de Advanced Persistent Threat (APT) e *exfiltração* de dados. A extensa revisão da literatura permitiu aos membros do Laboratório de Modelo Computacional para Transporte Aéreo (TransLab) a completa compreensão da dificuldade do problema abordado. Assim, recomendamos o uso e melhoria do modelo transformer para resolver os problemas mapeados. Tal como apostamos que as redes robustas em contextos dinâmicos são uma opção que precisa ser testada, devido às vantagens que pode trazer.

Também será necessário começar a aplicação de técnicas de Few-Shot Learning (FSL), ao menos nessa fase inicial do projeto, aonde o banco de dados ainda está pequeno. Além disso, é esperado em breve reunir uma quantidade razoável de meta-dados para

poder aplicar técnicas de *autoML*, previsão de performance e destarte, criar um dinâmico sistema de Data Lacking Protection System (DLPS), altamente adaptável em diversos contextos de ameaça cibernética.

Referências

[1] Sabir, Bushra, Faheem Ullah, M Ali Babar e Raj Gaire: *Machine learning for detecting data exfiltration: a review.* ACM Computing Surveys (CSUR), 54(3):1–47, 2021. xi, 2, 3, 7, 31, 32, 37

[2] Vaswani, Ashish, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser e Illia Polosukhin: *Attention is all you need.* Advances in neural information processing systems, 30, 2017. xi, 12, 13, 14, 46, 55

[3] Llorente, Dusthon, Mariana Ballesteros, Ivan DE JESUS Salgado Ramos e Jorge Isaac Chairez Oria: *Deep learning adapted to differential neural networks used as pattern classification of electrophysiological signals.* IEEE Transactions on Pattern Analysis and Machine Intelligence, 2021. xi, 9, 15, 16, 18, 53

[4] Rajalakshmi, E, N Asik Ibrahim e V Subramaniyaswamy: *A survey of machine learning techniques used to combat against the advanced persistent threat.* Em *International Conference on Applications and Techniques in Information Security*, páginas 159–172. Springer, 2019. 2

[5] Do Xuan, Cho, Mai Hoang Dao e Hoa Dinh Nguyen: *Apt attack detection based on flow network analysis techniques using deep learning.* Journal of Intelligent & Fuzzy Systems, 39(3):4785–4801, 2020. 2, 4

[6] Alshamrani, Adel, Sowmya Myneni, Ankur Chowdhary e Dijiang Huang: *A survey on advanced persistent threats: Techniques, solutions, challenges, and research opportunities.* IEEE Communications Surveys & Tutorials, 21(2):1851–1877, 2019. 2, 3

[7] Castro Martins, James de, Li Weigang, Luís Paulo Faina Garcia e Gabriel Alves Castro: *Dlps baseado em deep learning: Nova abordagem para detecção de exfiltração em hdfs.* Em *Anais do X Brazilian Workshop on Social Network Analysis and Mining*, páginas 229–240. SBC, 2021. 2

[8] Ullah, Faheem, Matthew Edwards, Rajiv Ramdhany, Ruzanna Chitchyan, M Ali Babar e Awais Rashid: *Data exfiltration: A review of external attack vectors and countermeasures.* Journal of Network and Computer Applications, 101:18–54, 2018. 2

[9] Fe-Fei, Li *et al.*: *A bayesian approach to unsupervised one-shot learning of object categories.* Em *proceedings ninth IEEE international conference on computer vision*, páginas 1134–1141. IEEE, 2003. 3, 8

[10] Leydesdorff, Loet e Henry Etzkowitz: *The triple helix as a model for innovation studies*. Science and public policy, 25(3):195–203, 1998. 4

[11] Zambrano, Patricio, Jenny Torres, Luis Tello-Oquendo, Rubén Jácome, Marco E Benalcázar, Roberto Andrade e Walter Fuertes: *Technical mapping of the grooming anatomy using machine learning paradigms: An information security approach*. IEEE Access, 7:142129–142146, 2019. 4

[12] Schindler, Timo: *Anomaly detection in log data using graph databases and machine learning to defend advanced persistent threats*. arXiv preprint arXiv:1802.00259, 2018. 4

[13] Tondon, Devika e Monika Khurana: *Security of big data in hadoop using aes-mr with auditing*. International Journal of Advanced Research in Computer Science and Software Engineering, 7(1):100–105, 2017. 5, 38

[14] Sadasivam, G Sudha, K Anitha Kumari e S Rubika: *A novel authentication service for hadoop in cloud environment*. Em *2012 IEEE International Conference on Cloud Computing in Emerging Markets (CCEM)*, páginas 1–6. IEEE, 2012. 5, 38

[15] Lin, Hsiao Ying, Shiuan Tzuo Shen, Wen Guey Tzeng e Bao Shuh P Lin: *Toward data confidentiality via integrating hybrid encryption schemes and hadoop distributed file system*. Em *2012 IEEE 26th International Conference on Advanced Information Networking and Applications*, páginas 740–747. IEEE, 2012. 5, 38

[16] Raman, Preeti, Hilmi Güneş Kayacık e Anil Somayaji: *Understanding data leak prevention*. Em *6th Annual Symposium on Information Assurance (ASIA'11)*, página 27. Citeseer, 2011. 7

[17] Brindha, T e RS Shaji: *An analysis of data leakage and prevention techniques in cloud environment*. Em *2015 International Conference on Control, Instrumentation, Communication and Computational Technologies (ICCICCT)*, páginas 350–355. IEEE, 2015. 7

[18] Alneyadi, Sultan, Elankayer Sithirasenan e Vallipuram Muthukkumarasamy: *A survey on data leakage prevention systems*. Journal of Network and Computer Applications, 62:137–152, 2016. 7

[19] Weigang, Li e Nilton Correia da Silva: *A study of parallel neural networks*. Em *IJCNN'99. International Joint Conference on Neural Networks. Proceedings (Cat. No. 99CH36339)*, volume 2, páginas 1113–1116. IEEE, 1999. 7, 8

[20] Duan, Ruixue, Dan Li, Qiang Tong, Tao Yang, Xiaotong Liu e Xiulei Liu: *A survey of few-shot learning: An effective method for intrusion detection*. Security and Communication Networks, 2021, 2021. 8, 22, 29

[21] Weigang, Li, Liriam Michi Enamoto, Denise Leyi Li e Geraldo Pereira Rocha Filho: *New directions for artificial intelligence: Human, machine, biological, and quantum intelligence*. Frontiers of Information Technology & Electronic Engineering, 23(6):984–990, 2022. 9

[22] Vanschoren, Joaquin: *Meta-learning: A survey.* arXiv preprint arXiv:1810.03548, 2018. 9, 21, 28

[23] Yin, Chengxiang, Jian Tang, Zhiyuan Xu e Yanzhi Wang: *Adversarial meta-learning.* arXiv preprint arXiv:1806.03316, 2018. 10

[24] Goldblum, Micah, Liam Fowl e Tom Goldstein: *Adversarially robust few-shot learning: A meta-learning approach.* Advances in Neural Information Processing Systems, 33:17886–17895, 2020. 10

[25] Grezes, Felix: *Reservoir Computing.* Tese de Doutoramento, Dissertation Submitted to the Graduate Faculty in Computer Science, The City . . . , 2014. 10

[26] Roy, Debaditya, Vangjush Komini e Sarunas Girdzijauskas: *Out-of-distribution in human activity recognition.* Em *2022 Swedish Artificial Intelligence Society Workshop (SAIS)*, páginas 1–10. IEEE, 2022. 11, 55

[27] Bhaduri, Aparna, Kun Qu, Carolyn S Lee, Alexander Ungewickell e Paul A Khavari: *Rapid identification of non-human sequences in high-throughput sequencing datasets.* Bioinformatics, 28(8):1174–1175, 2012. 11

[28] Kowsari, Kamran, Donald E Brown, Mojtaba Heidarysafa, Kiana Jafari Meimandi, Matthew S Gerber e Laura E Barnes: *Hdltex: Hierarchical deep learning for text classification.* Em *2017 16th IEEE international conference on machine learning and applications (ICMLA)*, páginas 364–371. IEEE, 2017. 11

[29] Nielsen, Michael A: *Neural networks and deep learning*, volume 25. Determination press San Francisco, CA, USA, 2015. 11

[30] Thickstun, John: *The transformer model in equations.* University of Washington, Tech. Rep, 2021. 12, 13

[31] Zhang, Xian Ming, Qing Long Han, Xiaohua Ge e Derui Ding: *An overview of recent developments in lyapunov–krasovskii functionals and stability criteria for recurrent neural networks with time-varying delays.* Neurocomputing, 313:392–401, 2018. 15

[32] Ma, Yao, Shilin Zhao, Weixiao Wang, Yaoman Li e Irwin King: *Multimodality in meta-learning: A comprehensive survey.* Knowledge-Based Systems, página 108976, 2022. 20, 26

[33] Wei, Wei, Chao Huang, Lianghao Xia, Yong Xu, Jiashu Zhao e Dawei Yin: *Contrastive meta learning with behavior multiplicity for recommendation.* Em *Proceedings of the Fifteenth ACM International Conference on Web Search and Data Mining*, páginas 1120–1128, 2022. 20, 26

[34] Kumar, Manoj, Dr Husain, Naveen Upreti, Deepti Gupta *et al.*: *Genetic algorithm: Review and application.* Available at SSRN 3529843, 2010. 20, 27

[35] Grefenstette, John J: *Genetic algorithms and machine learning.* Em *Proceedings of the sixth annual conference on Computational learning theory*, páginas 3–4, 1993. 20, 21, 27, 28

[36] Levitin, Gregory, Jacob Rubinovitz e Boris Shnits: *A genetic algorithm for robotic assembly line balancing*. European Journal of Operational Research, 168(3):811–825, 2006. 21, 28

[37] Horie, Kazuhiro e Bruce A Conway: *Genetic algorithm preprocessing for numerical solution of differential games problems*. Journal of guidance, control, and dynamics, 27(6):1075–1078, 2004. 21, 28

[38] Vas, Peter: *Artificial-intelligence-based electrical machines and drives: application of fuzzy, neural, fuzzy-neural, and genetic-algorithm-based techniques*, volume 45. Oxford university press, 1999. 21, 28

[39] Shapiro, Jonathan: *Genetic algorithms in machine learning*. Em *Advanced Course on Artificial Intelligence*, páginas 146–168. Springer, 1999. 21, 28

[40] Paulinas, Mantas e Andrius Ušinskas: *A survey of genetic algorithms applications for image enhancement and segmentation*. Information Technology and control, 36(3), 2007. 21, 28

[41] Zhao, Linchang, Zhaowei Shang, Ling Zhao, Anyong Qin e Yuan Yan Tang: *Siamese dense neural network for software defect prediction with small data*. IEEE Access, 7:7663–7677, 2018. 22, 28, 29

[42] He, Kaiming, Xiangyu Zhang, Shaoqing Ren e Jian Sun: *Deep residual learning for image recognition*. Em *Proceedings of the IEEE conference on computer vision and pattern recognition*, páginas 770–778, 2016. 22, 28

[43] Zagoruyko, Sergey e Nikos Komodakis: *Learning to compare image patches via convolutional neural networks*. Em *Proceedings of the IEEE conference on computer vision and pattern recognition*, páginas 4353–4361, 2015. 22, 28

[44] Fei-Fei, Li, Robert Fergus e Pietro Perona: *One-shot learning of object categories*. IEEE transactions on pattern analysis and machine intelligence, 28(4):594–611, 2006. 22, 29

[45] Wang, Yaqing, Quanming Yao, James T Kwok e Lionel M Ni: *Generalizing from a few examples: A survey on few-shot learning*. ACM computing surveys (csur), 53(3):1–34, 2020. 22, 29

[46] Pineda, José Octávio de Carvalho *et al.*: *A entropia segundo claude shannon: o desenvolvimento do conceito fundamental da teoria da informação*. 2006. 22, 29

[47] Das, Anirban, Min Yi Shen, Madhu Shashanka e Jisheng Wang: *Detection of exfiltration and tunneling over dns*. Em *2017 16th IEEE International Conference on Machine Learning and Applications (ICMLA)*, páginas 737–742. IEEE, 2017. 31

[48] Anton, Simon Duque, Suneetha Kanoor, Daniel Fraunholz e Hans Dieter Schotten: *Evaluation of machine learning-based anomaly detection algorithms on an industrial modbus/tcp data set*. Em *Proceedings of the 13th international conference on availability, reliability and security*, páginas 1–9, 2018. 31

[49] Anguita, Davide, Alessandro Ghio, Luca Oneto, Xavier Parra Perez e Jorge Luis Reyes Ortiz: *A public domain dataset for human activity recognition using smartphones.* Em *Proceedings of the 21th international European symposium on artificial neural networks, computational intelligence and machine learning*, páginas 437–442, 2013. 34, 35, 45, 57

[50] Anguita, Davide, Alessandro Ghio, Luca Oneto, Xavier Parra Perez e Jorge Luis Reyes Ortiz: *A public domain dataset for human activity recognition using smartphones.* Em *Proceedings of the 21th international European symposium on artificial neural networks, computational intelligence and machine learning*, páginas 437–442, 2013. 34, 35, 45, 57

[51] Lindemann, Benjamin, Timo Müller, Hannes Vietz, Nasser Jazdi e Michael Weyrich: *A survey on long short-term memory networks for time series prediction.* Procedia CIRP, 99:650–655, 2021. 35, 48

[52] Jang-Jaccard, Julian e Surya Nepal: *A survey of emerging threats in cybersecurity.* Journal of Computer and System Sciences, 80(5):973–993, 2014, ISSN 0022-0000. `https://www.sciencedirect.com/science/article/pii/S0022000014000178`, Special Issue on Dependable and Secure Computing. 37

[53] Cohen, Jason e Subatra Acharya: *Towards a more secure apache hadoop hdfs infrastructure.* Em *International Conference on Network and System Security*, páginas 735–741. Springer, 2013. 38

[54] Shekhawat, Hema, Samiksha Sharma e Anchal Pokharana: *An approach to secure data stored in hdfs.* 38

[55] Forouzan, Behrouz A: *TCP/IP protocol suite.* McGraw-Hill Higher Education, 2002. 44

[56] Kurose, James F e Keith W Ross: *Computer networking: A top-down approach edition.* Addision Wesley, 2007. 44

[57] Panchapagesan, Sankaran, Ming Sun, Aparna Khare, Spyros Matsoukas, Arindam Mandal, Björn Hoffmeister e Shiv Vitaladevuni: *Multi-task learning and weighted cross-entropy for dnn-based keyword spotting.* Em *Interspeech*, volume 9, páginas 760–764, 2016. 44

[58] Chai, Li, Jun Du, Qing Feng Liu e Chin Hui Lee: *A cross-entropy-guided measure (cegm) for assessing speech recognition performance and optimizing dnn-based speech enhancement.* IEEE/ACM Transactions on Audio, Speech, and Language Processing, 29:106–117, 2020. 44

[59] Sahraeian, Reza e Dirk Van Compernolle: *Cross-entropy training of dnn ensemble acoustic models for low-resource asr.* IEEE/ACM Transactions on Audio, Speech, and Language Processing, 26(11):1991–2001, 2018. 44

[60] Zhang, Zijun: *Improved adam optimizer for deep neural networks.* Em *2018 IEEE/ACM 26th International Symposium on Quality of Service (IWQoS)*, páginas 1–2. Ieee, 2018. 47