UNIVERSIDADE FEDERAL DE PERNAMBUCO
CENTRO DE INFORMÁTICA
SISTEMAS DE INFORMAÇÃO

ANTÔNIO PAULINO DE LIMA NETO

ESTUDO COMPARATIVO ENTRE MODELOS BASEADOS EM BERT NA CLASSIFICAÇÃO ESTÁTICA DE MALWARE

Recife
2023

ANTÔNIO PAULINO DE LIMA NETO

ESTUDO COMPARATIVO ENTRE MODELOS BASEADOS EM BERT NA CLASSIFICAÇÃO ESTÁTICA DE MALWARE

Trabalho apresentado ao Programa de Graduação em Sistemas de Informação do Centro de Informática da Universidade Federal de Pernambuco como requisito parcial para obtenção do grau de Bacharel em Sistemas de Informação.

Orientador: Cleber Zanchettin

Recife

2023

ANTÔNIO PAULINO DE LIMA NETO

ESTUDO COMPARATIVO ENTRE MODELOS BASEADOS EM BERT NA CLASSIFICAÇÃO ESTÁTICA DE MALWARE

Trabalho apresentado ao Programa de Graduação em Sistemas de Informação do Centro de Informática da Universidade Federal de Pernambuco como requisito parcial para obtenção do grau de Bacharel em Sistemas de Informação.

Recife, 04 de Abril de 2023

BANCA EXAMINADORA

_____

Prof. Cleber Zanchettin (Orientador)

UNIVERSIDADE FEDERAL DE PERNAMBUCO

_____

Prof. Divanilson Campelo (2º membro da banca)

UNIVERSIDADE FEDERAL DE PERNAMBUCO

AGRADECIMENTOS

Aos professores do Centro de Informática que tornaram possível a minha jornada e me ensinaram tantas habilidades e valores, em especial ao meu orientador.

A todos os meus colegas de estudo e trabalho pelo apoio nos muitos trabalhos, projetos e atividades que fizemos juntos.

A minha família e amigos por nunca deixar de acreditar na minha capacidade.

A minha noiva, por me dar forças para seguir em frente.

"Dê tempo a si mesmo. Ideias virão."

(Johnny Silverhand)

Resumo

Com o crescimento no número de incidentes que envolvem malware e o constante desenvolvimento de novas ameaças por atores maliciosos, existe uma necessidade cada vez maior de proteger sistemas informatizados e seus usuários. Neste trabalho, é apresentado um estudo comparativo sobre a utilização de modelos de linguagem baseados em BERT para classificar amostras de malware através da análise estática. Usando bases de dados destinadas ao *benchmarking* de modelos de análise estática, oito *transformers* foram comparados usando as métricas F1-Score, acurácia, área sob a curva ROC e coeficiente de correlação de Matthews para determinar seu desempenho na execução dessas tarefas. Após os testes, foi possível concluir que *transformers* possuem uma ligeira vantagem na separação entre as diferentes classes quando comparados a modelos de inteligência artificial mais clássicos.

Palavras-chave: Machine Learning, Malware, Classificação, Deep Learning, Análise estática, BERT, XLNet, RoBERTa, Benchmarking.

ABSTRACT

With the increasing number of malware incidents and the constant development of new threats by malicious actors, there is an ever-increasing need to protect computer systems and their users. This paper presents a comparative study of using language models based on BERT in classifying malware samples by static analysis. Using databases intended for benchmarking static analysis models, eight transformer models were compared using the metrics F1-Score, accuracy, the area under the ROC curve, and Matthews' correlation coefficient to determine their performance in the execution of these tasks. After testing, it was found that transformers have a slight advantage over others classic artificial intelligence models in separating the different classes.

Keywords: Machine Learning, Malware, Classificação, Deep Learning, Análise estática, BERT, XLNet, RoBERTa, Benchmarking.

Lista de Ilustrações

Lista de Tabelas

Lista de Abreviaturas e Siglas

ALBERT	A Light BERT
API	Application Programming Interface
AUC	Area Under Curve
BERT	Bidirectional Encoder Representations from Transformers
CCIP	Center for Cybersecurity and Infrastructure Protection
GPT	Generative Pretrained Transformer
LightGBM	Light Gradient Boosting Machine
k-NN	k-Nearest Neighbours
MCC	Matthews Correlation Coefficient
PE	Portable Executable
RoBERTa	Robustly Optimized BERT Pretraining Approach
RBM	Restricted Boltzmann Machine
SECBERT	Security BERT
SECRoBERTa	Security RoBERTa
SVM	Support Vector Machine
VIT	Visual Transformer

SUMÁRIO

1 INTRODUÇÃO

A cada ano, o número de ameaças cibernéticas aumenta e coloca em risco consumidores e empresas em todo o mundo. Novas famílias de malware surgem constantemente e com elas novas estratégias para evitar mecanismos de detecção e ferramentas de segurança. Embora sejam notáveis os esforços na identificação de ameaças, como o compartilhamento de indicadores de comprometimento entre grandes entidades da tecnologia, incidentes envolvendo malware são cada vez mais comuns. Em 2021, foram detectados em média 380 mil novos arquivos maliciosos por dia, sendo a grande maioria deles (91%) arquivos Portable Executable (PE) do Windows (KASPERSKY, 2022). Ainda segundo a SonicWall (2022), apenas no primeiro semestre de 2022, foram detectados mais de 2.8 bilhões de incidentes envolvendo malware, um aumento de 11% em relação ao mesmo período do ano anterior.

Embora seja difícil estimar o prejuízo exato causado por malware, uma vez que muitas organizações não divulgam dados relacionados a incidentes cibernéticos (COULTER, 2019), uma pesquisa da Cybersecurity Ventures estima que os prejuízos causados por crimes cibernéticos sejam de US$ 8 trilhões em 2023, podendo chegar ao valor anual de US$ 10.5 trilhões em 2025 (MORGAN, 2022).

Ainda, segundo o Cybersecurity Workforce Study (INTERNATIONAL INFORMATION SYSTEM SECURITY CERTIFICATION CONSORTIUM, 2022), desde 2022 o campo da cibersegurança tem um déficit de 3.4 milhões de cargos, apesar do crescimento no número de profissionais no mesmo ano.

Existem diversos desafios para o futuro da detecção de malware, sendo um deles a automação do processo de criação de novas amostras. Determinadas ferramentas, que grupos criminosos podem alugar ou vender para atores menos treinados, são capazes de desenvolver e distribuir milhares de novas amostras de malware todos os dias e permitir que mesmo desenvolvedores com pouco conhecimento técnico criem seus próprios malwares (ARIFFIN, OMAR e SIHWAIL 2018).

Além disso, as funcionalidades e estrutura de programas maliciosos continuam mudando e se tornando cada vez mais complexas, com o emprego de novas técnicas de ofuscação que visam dificultar o processo de análise e detecção de ameaças (ARIFFIN, OMAR e SIHWAIL 2018).

Sabendo dessa necessidade e tendo em mente que problemas de análise e classificação de malwares podem ser encarados como problemas de classificação, este trabalho tem como objetivo comparar modelos de *transformers* baseados em BERT na análise estática e classificação de programas maliciosos

Os *transformers* emergiram como uma classe promissora de modelos de aprendizado profundo para análise estática e classificação de programas maliciosos devido à sua capacidade notável de entender e classificar sequências de dados. Sua arquitetura de atenção permite que eles considerem todas as posições na entrada simultaneamente, permitindo um processamento eficiente e uma representação mais rica do contexto. Além disso, *transformers* pré-treinados possibilitam a transferência de conhecimento a partir de grandes conjuntos de dados, o que contribui para uma detecção de malware mais precisa e generalizada e com base nisso se tornaram alvo de estudo neste trabalho de graduação.

1.1 OBJETIVOS

1.1.1 Gerais

Este trabalho tem como objetivo comparar o desempenho de diferentes modelos de *transformers* baseados no Bert em tarefas de classificação estática de malware.

1.1.2 Específicos

- Definição de bases de dados para comparação dos modelos;
- Aplicação dos modelos na classificação das amostras que compõem as bases de dados;
- Comparação do desempenho dos modelos a partir das métricas de avaliação escolhidas;

2 REFERENCIAL TEÓRICO

Este capítulo tem como objetivo explicar os principais conceitos abordados neste trabalho, assim como recapitular um pouco da história de sistemas do malware, de mecanismos de detecção e da inteligência artificial e processamento de linguagem natural.

2.1 MALWARE

Malware, abreviação do inglês *malicious software*, é o termo que define qualquer software que tem como objetivo causar problemas a computadores ou seus usuários (TAHIR, 2018). Os malwares são divididos em subcategorias a partir de suas capacidades e da forma como atuam para prejudicar os sistemas que comprometem, dentre as categorias conhecidas estão:

- Vírus: Gill (2022) descreve o vírus como um tipo de malware capaz de se espalhar para outros computadores, comumente através de arquivos que, quando abertos, são responsáveis por iniciar a execução do programa malicioso;
- Spyware: o spyware, abreviação do inglês *spy software*, é descrito por Gill (2022) como um tipo de malware que tem como objetivo coletar informação do usuário através do monitoramento do dispositivo. Esses programas podem ter alvos variados, dentre eles senhas, arquivos sensíveis no dispositivo do usuário e números de cartão de crédito;
- *Downloader*: também chamados de "*droppers*", *downloaders* são malwares projetados para baixar e executar outros malwares no dispositivo da vítima, segundo TROJAN (2020). Esse tipo de ameaça é comumente distribuído através de páginas e e-mails maliciosos;
- Ransomware: do inglês, *ransom software*. Segundo AL-HAWAWREH, HARTOG e SITNIKOVA (2019), malwares do tipo ransomware têm como objetivo impedir um usuário de acessar um dispositivo, sistema ou arquivos dentro de um sistema até que um valor de resgate, comumente cobrado em criptomoedas, seja pago. A subcategoria mais conhecida desse malware é o crypto-ransomware, que usa criptografia para corromper os arquivos da vítima.
- *Backdoor*: ou "porta dos fundos", em português, são malwares desse tipo são implementados para permitir que atacantes tenham acesso remoto ao sistema alvo, comumente a partir da execução de comandos nos dispositivos comprometidos (GILL, 2022).

2.1.2 DETECÇÃO DE MALWARE

Sistemas de detecção de malware, comumente chamados de antivírus, são, segundo Koret e Bachaalany (2015), programas criados para impedir que um computador seja infectado a partir da detecção de *software* malicioso [...] e, quando apropriado, remoção do malware e desinfecção do computador (tradução nossa).

O primeiro malware que se tem registro, o *Creeper*, foi um malware do tipo worm criado de forma experimental pelo pesquisador Bob Thomas em 1971, ele era capaz de se propagar de forma autônoma pela ARPANET (CHEN e ROBERT, 2004), exibindo a mensagem “eu sou o *Creeper*: pegue-me se for capaz” (MELTZER e PHILLIPS, 2009, tradução nossa) nos terminais infectados. O surgimento desse malware foi acompanhado pela criação do *Reaper*, um software que, de forma análoga ao *Creeper*, também se propagava de forma autônoma, mas que tinha como função removê-lo dos dispositivos infectados (MELTZER e PHILLIPS, 2009).

Embora o *Reaper* possa ser considerado o primeiro software antivírus de que se tem registro (MELTZER e PHILLIPS, 2009), o termo se popularizou apenas na década de 80 com o lançamento das primeiras soluções antivírus comerciais, como o G DATA AntiVirus Kit, um dos primeiros softwares antivírus comerciais desenvolvido para proteção do Atari ST (G DATA SOFTWARE, 2017) e e o Dr. Solomon's Anti-Virus Toolkit (JACKSON, 1989).

Desde então, os softwares antivírus evoluíram de simples *scanners* de linha de comando que tentam identificar padrões maliciosos em arquivos para ferramentas complexas capazes de monitorar quaisquer arquivos criados, modificados ou acessados pelo sistema operacional, além de contar com *firewalls* que monitoram como programas usam a Internet (KORET e BACHAALANY, 2015) e empregar as mais variadas técnicas de detecção que vão desde regras heurísticas a modelos de inteligência artificial (PÉREZ-SÁNCHEZ e PALACIOS, 2022).

Atualmente existem duas principais estratégias usadas na análise de arquivos potencialmente maliciosos, são elas as análises estáticas e dinâmicas (CHAKKARAVARTHY, SANGEETHA e VAIDEHI, 2019).

2.1.2.1 Análise estática

A análise estática consiste na inspeção de programas potencialmente maliciosos sem que eles sejam executados (ASLAN e SAMET, 2020). Técnicas de análise estática usam características do arquivo executável, como strings presentes no código-fonte do programa, sequências de *bytes* e importações de bibliotecas para determinar se um software é malicioso. (GANDOTRA, BANSAL e SOFAT, 2014).

Para dificultar a análise estática, desenvolvedores de malware empregam técnicas de encapsulamento e criptografia usando programas como *crypters* e *packers* (ABOAOJA et al, 2022), esses comumente, transformam os programas maliciosos em binários autocontidos, ocultando informações como tamanhos das estruturas de dados e chamadas de API. Ainda segundo Gandotra, Bansal e Sofat (2014), a constante evolução de técnicas de evasão sendo adotadas por desenvolvedores de malware foi o que motivou o surgimento de técnicas de análise dinâmica.

2.1.2.2 Análise dinâmica

A análise dinâmica, conhecida também como análise de comportamento, consiste na execução do malware em um ambiente controlado, como uma máquina virtual ou em soluções de *sandbox*, e observação de seu comportamento (GANDOTRA, BANSAL e SOFAT, 2014). O comportamento de um malware é definido pelas ações executadas por ele durante sua execução no sistema operacional (GRÉGIO, 2012). Dentre as técnicas usadas para análise dinâmica estão o monitoramento de chamadas de funções e do fluxo de informação do programa (GANDOTRA, BANSAL e SOFAT, 2014).

Embora a análise dinâmica seja mais efetiva que a estática, uma vez que é baseada no comportamento natural do malware, ela demanda mais tempo e recursos computacionais para ser executada, o que traz problemas para a escalabilidade dessas soluções, além das diferenças entre os ambientes controlados onde os malwares são analisados e daqueles onde serão executados o que podem levar a comportamentos artificiais e diferença de resultados de análise segundo (GANDOTRA, BANSAL E SOFAT, 2014). Por esses motivos, os testes

realizados neste trabalho possuem como foco a análise de características estáticas do malware, como descrito adiante na subseção Dataset do capítulo Metodologia.

2.2 APRENDIZAGEM DE MÁQUINA

O termo “aprendizagem de máquina” foi criado em 1959 por Arthur Samuel e é definido como um ramo da ciência da computação e inteligência artificial que tem como foco principal o uso de dados e algoritmos para simular a forma como humanos aprendem, melhorando gradualmente sua precisão (IBM, 2020).

Algoritmos de aprendizagem de máquina visam o reconhecimento de padrões a partir de um conjunto de dados extraídos de um dado problema. Esses algoritmos são divididos entre algoritmos de classificação, que visam a classificação de amostras desconhecidas através de padrões aprendidos a partir dos dados conhecidos, e algoritmos de regressão, que usam os padrões conhecidos para prever resultados de eventos relacionados ao problema.

Uma vez que este trabalho tem como objetivo a classificação de amostras de malware, a seção a seguir descreve de forma resumida os algoritmos de classificação e sua evolução. Embora algoritmos de regressão e outras técnicas de aprendizagem de máquina também possam ser usados na detecção de malware, este trabalho tem como objeto de estudo apenas algoritmos de classificação, pois estão mais associados ao objetivo proposto e serão o foco das análises.

2.2.1 Classificação

Segundo Neelamegam e Ramaraj (2013), “a classificação é uma técnica de mineração de dados que visa prever a quais grupos instâncias de dados pertencem”. Para o cumprimento dessa tarefa são empregados algoritmos criados especialmente com esse objetivo. Segundo Gama e Brazdil (1995), “é difícil identificar um algoritmo de classificação que tenha bom desempenho em todas as tarefas”.

Em 1943, o Perceptron, base das redes neurais atuais, foi proposto pelos pesquisadores Warren McCulloch e Walter Pitts (1943), embora tenha sido implementado pela primeira vez apenas em 1958 por Frank Rosenblatt (1958). O Perceptron é um classificador binário que recebe uma entrada e, a partir de uma função linear e um vetor de pesos, calcula se ela

pertence a um dado grupo. Embora inicialmente tenha sido considerado promissor, o Perceptron como foi inicialmente concebido foi deixado de lado por, dentre outros motivos, sua incapacidade de solucionar problemas de classificação não-linearmente separáveis (MINSKY e PAPERT, 1969).

Na década de 60 e início dos anos 70, as árvores de decisão se tornaram algoritmos populares na resolução de problemas de classificação (RAO e MITRA, 1972). Ainda na década de 60, algoritmos estatísticos mais antigos foram adotados na resolução de problemas de classificação, como a Análise de Discriminante Linear, a Regressão Logística (BERKSON, 1944) e o algoritmo k-NN (do inglês. *k-Nearest Neighbours*) (COVER e HART, 1967).

Outros algoritmos de classificação populares até hoje são o algoritmo Naive Bayes, que utiliza o teorema de Bayes para classificação (GOOD, Isidore Jacob, 1950) e o algoritmo de Máquinas de Vetores de Suporte (do inglês, *Support Vector Machine*, ou SVM).

Nas décadas de 80 e 90, no entanto, o campo de aprendizagem de máquina redescobriu as redes neurais com a introdução do algoritmo de backpropagation (RUMELHART, HINTON e WILLIAMS, 1986). E, a partir dos anos 2000, o aumento do poder computacional e o desenvolvimento de novos algoritmos de aprendizagem profunda levaram a avanços significativos em tarefas de classificação de imagem, fala e texto, especialmente com a recente introdução dos modelos de *transformers* (VASWANI et al, 2017), popularizados por soluções de processamento de linguagem natural como o BERT (CHANG et al, 2018) e os modelos GPT (NARASIMHAN et al, 2018), mas aplicados ao cumprimento de diversas tarefas, como a classificação (YANG et al, 2022) e geração de imagens (CHEN et al, 2022).

2.2.2 Aprendizagem de máquina na detecção de malware

A detecção de malware tem sido uma preocupação desde o surgimento dos primeiros vírus de computador nas décadas de 1970 e 1980. Embora inicialmente a detecção de malware empregada por softwares antivírus tenha se baseado em regras simples, como a busca de sequências de bytes específicas em arquivos (RAD, MASROM e IBRAHIM, 2011), os primeiros experimentos envolvendo a detecção de malware a partir de técnicas de machine learning são da década de 90 (GUINIER, 1991).

A aplicação de algoritmos de aprendizagem de máquina na detecção e classificação de malware tem se popularizado na última década, sendo o surgimento de bases de dados de

malware públicas (BRIGUGLIO, ELMILIGI e SAAD, 2019), como VirusTotal e VirusShare, e o aumento do poder de processamento dos computadores os principais motivos por trás dessa popularização (GIBERT, MATEU e PLANES, 2020).

Desde então, soluções empregando os mais variados modelos vêm sendo desenvolvidas por pesquisadores. Modelos como o de Carvalho, Chan e Hassen (2017), por exemplo, que usam *random forests*, alcançam altos níveis de precisão na detecção de famílias de malware a partir da análise de atributos estáticos das amostras, chegando a atingir 99.2% de acurácia.

No campo de detecção por análise dinâmica, experimentos baseados na análise de comportamento malicioso a partir de chamadas de API, modificações de registro e tráfego de rede chegam a níveis de precisão de 98.0% (CAVAZOS, LA ROSA e KILGALLON, 2017). Diversos modelos de detecção híbridos também foram propostos nos últimos anos. Estes visam aumentar a qualidade e precisão de sistemas de detecção de malware combinando técnicas de análise estática e dinâmica para criar sistemas de detecção mais robustos (BRIGUGLIO, ELMILIGI e SAAD 2019).

Atualmente, os principais softwares antivírus comerciais empregam algoritmos de aprendizagem de máquina na detecção de malware (SHIVANI, 2021).

2.3 PROCESSAMENTO DE LINGUAGEM NATURAL

Chopra, Prashar e Sain (2013) definem o Processamento de Linguagem Natural (NLP, na sigla em inglês) como uma subárea da inteligência artificial e linguística dedicada ao desenvolvimento de formas de os computadores entenderem a linguagem humana. A área tem origem ainda em 1950, quando Alan Turing propôs o Jogo da Imitação, agora conhecido como teste de Turing, como uma subárea da inteligência (TURING, 1950).

Ainda na década de 50, pesquisadores começaram a explorar a possibilidade de usar computadores para traduzir textos entre idiomas (HUTCHINS, 2005). No entanto, as primeiras tentativas de tradução automática eram limitadas e produziam resultados imprecisos (CHAPMAN, MACHADO e NADKARNI, 2011).

Já nas décadas de 60 e 70, as pesquisas em NLP passaram a focar na análise de textos em inglês (WINOGRAD, 1971). A maioria desses primeiros sistemas usava regras gramaticais codificadas manualmente para analisar a estrutura de frases e extrair informações

significativas. Esses sistemas eram complexos e geralmente não conseguiam entender o significado completo do texto (CHAPMAN, MACHADO e NADKARNI, 2011).

Ainda nesse período surgiram os primeiros chatbots, programas cuja função é simular uma conversação entre o computador e um usuário, como o ELIZA (WEIZENBAUM, 1966).

Foi a partir do fim da década de 80 e início dos anos 90, com a introdução de algoritmos de aprendizagem de máquina e de grandes corpos de texto (corpora), que surgiram os primeiros modelos de NLP probabilísticos, sendo estes destinados a tradução automática de texto (BROWN et al, 1988).

Nos anos 1990 e 2000, com o avanço do poder computacional e a disponibilidade de grandes conjuntos de texto, viabilizaram o desenvolvimento de modelos de NLP mais complexos e a realização de tarefas mais sofisticadas, como a análise de sentimentos (TURNEY, 2002) e a extração de informações.

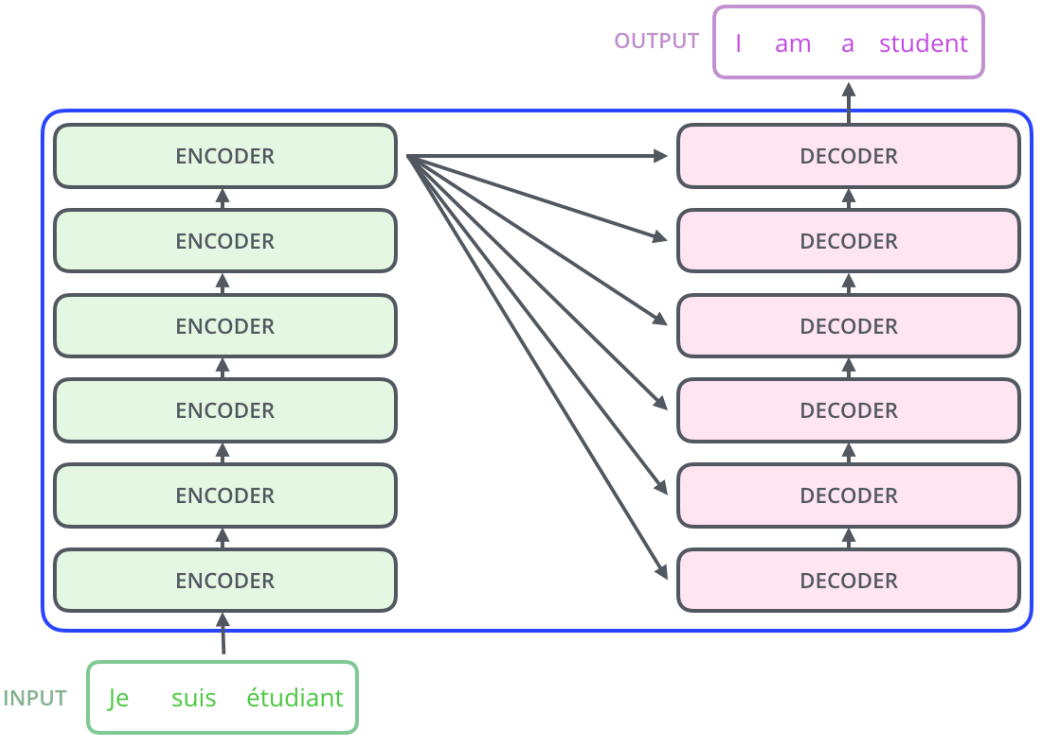
Nos últimos anos, com a popularização da aprendizagem profunda e a disponibilidade de grandes conjuntos de dados, a ênfase no NLP mudou para o treinamento de modelos pré-treinados baseados em *Transformers* (VASWANI, 2017) e transferência de aprendizado (SINCLAIR, 2021). Modelos como o BERT (*Bidirectional Encoder Representations from Transformers*) (CHANG et al, 2018) e o GPT (*Generative Pretrained Transformer*) (NARASIMHAN et al, 2018) têm se tornado a base para uma ampla variedade de soluções NLP personalizadas.

2.3.1 TRANSFORMERS

Propostas pela primeira vez em 2017, *transformers* são redes neurais que empregam mecanismos de autoatenção (do inglês, *self-attention*) no processamento de sequências de entrada (VASWANI et al, 2017). Esse mecanismo permite que o modelo se concentre nas partes mais importantes da sequência de entrada, com base em sua relevância para a tarefa em questão, para produzir uma saída mais precisa.

Essa arquitetura implementa camadas que atuam como modelos de *encoders* e *decoders*, cada uma com seu próprio mecanismo de autoatenção (ALAMMAR, 2018), sendo popularmente adotada em tarefas de tradução automática. A Figura 1 exemplifica a arquitetura de um modelo Transformer.

Figura 1 - exemplo de uma arquitetura Transformer aplicada a um problema de tradução automática, ilustrando as etapas de processamento e a interação entre elas. A arquitetura inclui um encoder para capturar informações contextuais da entrada e um decoder para gerar a tradução, trabalhando juntos para melhorar a qualidade do resultado.



Fonte: Alammar, 2018.

Esses modelos são pré-treinados em grandes conjuntos de dados antes de passarem por um processo de ajuste fino que visa prepará-los para desempenhar tarefas específicas. Isso permite que a rede aprenda representações de alta qualidade dos dados, o que pode melhorar significativamente o desempenho nessas tarefas (HENDRYCKS, LEE e MAZEIKA, 2019). Graças a esse pré-treinamento, esses modelos podem ser usados para tarefas de classificação específicas, mesmo quando os dados de treinamento são limitados. Essa capacidade é conhecida como transferência de aprendizado (ou *transfer learning*) e pode ser útil em muitos casos práticos (TORREY e SHAVLI, 2010).

2.3.2 BERT

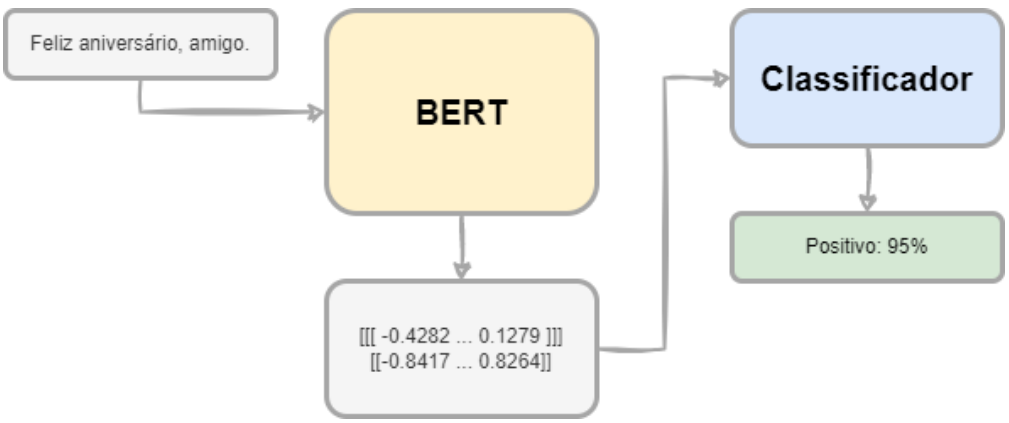
O BERT, sigla que significa "*Bidirectional Encoder Representations from Transformers*", é um modelo de linguagem pré-treinado baseado em *transformers* que foi projetado para realizar uma ampla variedade de tarefas de processamento de linguagem natural (CHANG et al, 2018). O BERT é bidirecional, o que significa que, quando tenta entender o significado de uma palavra, o modelo leva em consideração as palavras à esquerda e à direita dela.

Diferente da arquitetura padrão de *Transformers*, descrita na seção anterior, o BERT apenas utiliza a pilha de Encoders da arquitetura, o que significa que, ao receber uma sequência de palavras como entrada, o BERT produz um vetor de saída que representa essa sequência (ALAMMAR, 2018).

O processo de treinamento do BERT se deu através da tarefa de preenchimento de lacunas, do inglês *masked language modeling*, que consiste em fornecer como entrada ao modelo uma frase com uma das palavras ausentes e receber como saída a palavra que falta na frase.

Segundo Alammar (2018), o objetivo mais simples que pode ser alcançado com o BERT é o de classificação textual, isso pode incluir tarefas de análise de sentimento e detecção de spam. Para isso, o modelo é usado em conjunto com um algoritmo de classificação que recebe como entrada a saída do BERT, como demonstra a Figura 2.

Figura 2 - Exemplo de arquitetura de classificação usando o modelo BERT. A informação de entrada é codificada usando os *embeddings* do modelo BERT e depois classificada de acordo com as classes usadas no treinamento do modelo.



Fonte: O autor (2023)

Além da classificação textual, o BERT é projetado para ser facilmente adaptável a novas tarefas, permitindo que ele seja usado como base para a criação de outros modelos de linguagem (ALAMMAR, 2018).

Devido a suas características únicas e capacidade de lidar com grandes volumes de dados, o BERT é um candidato promissor para tarefas de classificação de malware. Sua capacidade de processar sequências de dados e usar mecanismos de autoatenção para identificar as relações entre os elementos da sequência é crucial para a análise de códigos maliciosos. Além disso, a transferência de aprendizado permite que o BERT seja pré-treinado em grandes conjuntos de dados não rotulados, tornando-o ideal para lidar com dados de treinamento limitados e detecção de malware em tempo real. Essas vantagens tornam o BERT, e demais *transformers*, uma das ferramentas mais promissoras para a detecção e classificação de malware.

3 TRABALHOS RELACIONADOS

Esta seção tem como objetivo apresentar outros trabalhos relacionados a detecção de malware.

3.1 Classificação de malware em ambientes Windows

Alsmadi e Alqudah (2021), catalogaram diversas técnicas de detecção de malware, dentre elas técnicas de detecção estáticas, dinâmicas e híbridas. Segundo sua pesquisa, modelos de detecção baseados em imagens que usam técnicas de aprendizagem de máquina e redes profundas possuem o melhor desempenho e eficiência computacional.

Essa é a abordagem usada no modelo desenvolvido por Huang et al. (2021), que se baseia no VGG16 e usa redes neurais convolucionais para abordar a detecção de malware como um problema de visão computacional. A partir da extração de características do malware, como informações de frequência de bytes, e visualização dessas características como canais RGB, o modelo foi capaz de alcançar valores de acurácia de 94.7%.

Por outro lado, Demetrio et al. (2021), analisaram um conjunto de ataques adversariais que podem ser empregados por atores maliciosos para enganar algoritmos de detecção baseados em aprendizagem de máquina. Neste trabalho, os pesquisadores conseguiram dissuadir diversos algoritmos de detecção , dentre eles algoritmos baseados em redes neurais convolucionais, conseguindo taxas de sucesso entre 60 e 100%, dependendo do algoritmo e do ataque.

Já Santos et al. (2013), propõem um modelo de detecção que visa, a partir do código assembly de um arquivo executável potencialmente malicioso, determinar sua legitimidade a partir da frequência de códigos de operação (do inglês *opcode*) no arquivo. Essa abordagem, no entanto, pode ser ineficiente frente ao uso de softwares *packer* (ALSMADI e ALQUDAH, 2021).

Azeez et al. (2021), por sua vez, propuseram um trabalho de classificação usando comitês de classificações (do inglês, *ensemble learning*) a partir de uma base de dados originalmente constituída por mais de 19 mil amostras de malware compostas por 77 características estáticas extraídas das amostras. Esses modelos foram capazes de obter níveis de acurácia de 99.24% com 0.98 de F-1 Score.

O modelo de detecção por assinatura usado pela ferramenta de código-aberto YARA, segundo Jaramillo (2018), está presente em diversas soluções antivírus comerciais, como o Nessus, ferramenta de detecção da Tenable.

3.1.1 Classificação de malware a partir de chamadas de API

O modelo de classificação proposto por Nikolopoulos e Polenakis (2017), cria estruturas em grafo a partir das chamadas de API feitas pelo malware e os classifica a partir da semelhança entre essas estruturas. Esse modelo foi capaz de atingir uma taxa de detecção de 94.7%.

Outro modelo, proposto por Assegie (2021), utiliza o algoritmo KNN para classificar arquivos potencialmente maliciosos a partir de sequências de chamadas de API. Os experimentos provaram que, com valor de *k* igual a 3, o modelo é capaz de atingir um nível de acurácia de 98.17%.

Já a partir de uma abordagem de detecção dinâmica, experimentos com os algoritmos de aprendizagem de máquina XG Boost e Random Forests, assim como com redes neurais profundas, foram capazes de obter nível de acurácia de 96.3% (KANG e WON, 2020, apud ALSMADI e ALQUDAH, 2021).

Galal, Mahdy e Atiea (2016), propuseram um modelo de extração de características dos malwares a partir da observação em tempo de execução de chamadas feita a API do Windows. Utilizando esse método de extração, foi possível atingir um nível de acurácia de 97.19% usando árvores de decisão. Essa técnica, no entanto, é ineficiente na análise de amostras que dependam de eventos externos, como o contato com um servidor de comando e controle.

Ye et al. (2018), por sua vez, adotou um modelo construído por um AutoEncoder e camadas de máquinas de Boltzmann restritas (RBM, da sigla em inglês), para classificar arquivos PE a partir das chamadas de API feitas por eles. Embora tenha sido capaz de atingir 98% de acurácia na tarefa de classificação binária de malware, isto é, se os arquivos são maliciosos ou não, essa abordagem possui um custo computacional considerável, uma vez que depende da extração dinâmica de chamadas de API.

3.2 Transformers e classificação de malware

Rahali e Akhloufi (2021), descrevem um modelo de classificação baseado no BERT que visa diferenciar softwares maliciosos e benignos desenvolvidos para Android. Apelidado de MalBERT, o modelo foi treinado a partir de características extraídas em programas disponibilizados no Androzoo, uma base de dados pública de artefatos maliciosos para Android. Neste trabalho, os pesquisadores trataram o problema tanto como um problema binário de classificação de texto, quanto um problema de múltiplas categorias, atingindo níveis de acurácia de cerca de 97% e 91%, nos respectivos problemas.

Ainda na voltado a detecção de malwares para Android, Seneviratne et al. (2022) desenvolveram um modelo baseado no *Visual Transformer* (VIT) capaz de atingir um nível de acurácia de 97% na classificação binária de amostras.

Já na classificação de malware para sistemas Windows, tópico deste trabalho, o modelo desenvolvido por Ghourabi (2022), que combina a arquitetura do BERT com o *Light Gradient Boosting Machine* (LightGBM), um framework de gradient boosting de código aberto mantido pela Microsoft, foi usado na construção de um sistema de detecção de malware, obtendo um nível de acurácia de 99% em tarefas de classificação binária.

4 METODOLOGIA

Este trabalho foi conduzido a partir de uma estratégia de pesquisa experimental, uma vez que seu objetivo é avaliar o desempenho de modelos Transformer na classificação de amostras de malware, sendo ele dividido nas seguintes etapas: análise de dados, investigação dos modelos e experimentação.

Os experimentos descritos neste trabalho foram conduzidos usando a base de dados *Benchmark Static API Call Datasets for Malware Family Classification* (Düzgün et al, 2022), disponibilizada pelo *Center for Cybersecurity and Infrastructure Protection* (CCIP) da Universidade Kadir Has, na Turquia. A base é composta por listas de chamadas de API coletadas a partir de análise estáticas de amostras de malware disponíveis nas plataformas VirusSample e VirusShare, plataformas abertas que visam o compartilhamento de malware malicioso para pesquisadores de segurança.

Esta base de dados foi escolhida por ter sido desenvolvida visando a avaliação de desempenho de modelos de detecção estáticos. A pesquisa de Düzgün et al. (2022) traz métricas comparativas de modelos de classificação quando aplicados aos dados da base. Estas métricas, por sua vez, são utilizadas para comparar o desempenho dos modelos avaliados neste experimento, como descreve a subseção Diretrizes dos experimentos, adiante.

4.1 Tratamento dos dados

Ao todo, a base contém dados de 9.795 amostras de malware da plataforma VirusSample e 14.616 da plataforma VirusShare, classificados como um de 15 possíveis tipos de malware distribuídos de acordo com a tabela a seguir:

Tabela 1 - Distribuição de amostras por categoria

Malware Family	VirusShare	VirusSample
Trojan	8.919	6.153
Vírus	2.490	2.367
Adware	908	222
Undefined	577	N/A
Worms	524	441
Backdoor	510	447
Downloader	218	31
Agent	165	102
Ransomware	115	10
Riskware	85	4
Spyware	45	11
Dropper	40	4
Crypt	10	2
Keylogger	7	1
Rootkit	3	N/A

Fonte: Düzgün et al, 2022.

A primeira etapa da fase de tratamento de dados seguiu os passos de regularização propostos por Düzgün et al. (2022). Isto inclui a remoção de amostras sem classificação definida da base de dados, isto é, marcadas com o rótulo *Undefined*. Em seguida, houve também a remoção de amostras de malware pertencentes a categorias com menos de 100 representantes na base, resultando na distribuição apresentada na tabela a seguir:

Tabela 2 - distribuição das amostras após regularização das base de dados

Malware Family	VirusShare	VirusSample
Trojan	8.919	6.153
Vírus	2.490	2.367
Adware	908	222
Worms	524	441
Backdoor	510	447
Downloader	218	N/A
Agent	165	102
Ransomware	115	N/A

Fonte: Düzgün et al, 2022.

Como proposto por Düzgün et al. (2022), cada um dos modelos foi testado em duas versões de ambas as bases de dados, sendo uma delas a versão representada pela distribuição na Tabela 2 e a outra uma versão balanceada por subamostragem, apresentada na Tabela 3, que consistiu em limitar o máximo de amostras por classe em 300.

Tabela 3 - distribuição de amostras nas versões balanceadas da base de dados

Malware Family	**VirusShare**	**VirusSample**
Trojan	300	300
Vírus	300	300
Adware	300	222
Worms	300	300
Backdoor	300	300
Downloader	300	N/A
Agent	165	102
Ransomware	115	N/A

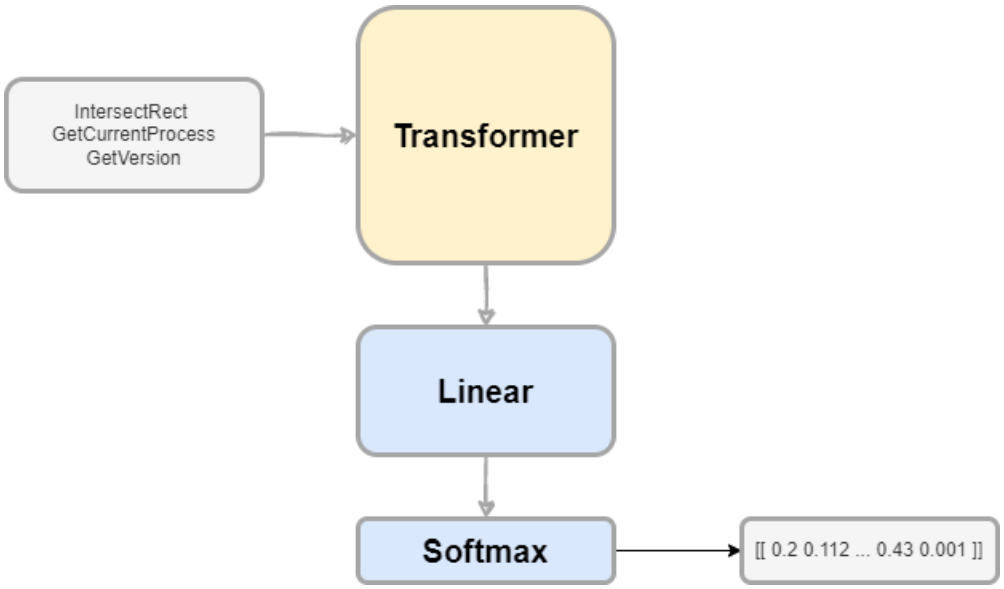
Fonte: Düzgün et al, 2022.

As bases foram então divididas em conjuntos de treinamento, teste e validação, compostos por 80%, 10% e 10% das amostras, respectivamente.

4.2 Diretrizes dos experimentos

Conforme detalha a seção 2.3.2 deste trabalho, modelos de classificação que usam o BERT podem ser construídos a partir da adição de uma camada linear de classificação ao modelo. Essa camada recebe como entrada a saída produzida pelo BERT, produzindo por sua vez um vetor de probabilidades de aquela amostra pertencer a cada um dos possíveis grupos de malware a partir da função de ativação Softmax. Para cálculo de perda, foi escolhida a função de entropia cruzada (*cross entropy*), uma vez que é a mais adequada para tarefas de classificação com múltiplas classes (BROWNLEE, 2019). A figura a seguir resume a arquitetura adotada para os modelos usados nos experimentos deste trabalho:

Figura 3 - Exemplo de arquitetura dos modelos de classificação adotados neste trabalho usando uma camada linear para classificação dos outputs dos *Transformers*.



Fonte: O autor (2023)

Com esses critérios em mente, foi realizada uma busca por modelos de transformer usando os seguintes critérios:

- O trabalho que descreve o modelo deveria estar em português ou inglês;
- Apenas trabalhos disponibilizados de forma gratuita foram selecionados;
- Trabalhos cujos código-fonte não foram disponibilizados não foram incluídos pois não permitiriam uma comparação justa entre eles.

Os oito modelos de linguagem escolhidos para comparação nos experimentos a partir dessa busca foram: BERT, ALBERT, DistilBERT, RoBERTa, XLNet, SecBERT, SecRoBERTa e cyBERT.

Cada um desses modelos oferece características únicas e aprimoramentos em relação aos seus antecessores. O RoBERTa, por exemplo, traz melhorias no processo de treinamento em relação ao BERT (LIU et al, 2019). ALBERT e DistilBERT, por sua vez, focam na redução de recursos computacionais sem comprometer significativamente a precisão (LAN et al, 2019; SANH et al., 2019). Já o XLNet aborda as limitações do BERT com a modelagem de linguagem permutada (YANG et al, 2019). Por sua vez, SecBERT e SecRoBERTa são adaptações específicas dos modelos BERT e RoBERTa para o domínio da segurança cibernética (SECBERT, 2020), o que pode trazer um impacto na sua capacidade para lidar com malwares. Por fim, como o cyBERT é projetado especificamente para a detecção de

anomalias em logs (RICHARDSON, 2019), acaba por ser um candidato promissor a essa tarefa. Avaliar o desempenho desses modelos em análise estática e classificação de malwares permite identificar qual deles apresenta a melhor combinação de precisão, escalabilidade e eficiência computacional para enfrentar o cenário em constante evolução das ameaças cibernéticas. A seguir uma breve descrição sobre cada um deles:

- BERT: O BERT é um modelo de aprendizado profundo baseado em Transformers, desenvolvido pelo Google AI Language. Sua principal inovação é o uso da atenção bidirecional, permitindo que ele considere o contexto de palavras à esquerda e à direita em uma frase. Isso melhora a capacidade do modelo de entender e gerar linguagem natural (CHANG et al, 2018). Mais detalhes sobre o modelo podem ser encontrados na seção **Referencial Teórico**.
- DistillBERT: Em 2019, pesquisadores da Hugging Face em 2019 apresentaram o DistilBERT, uma versão menor e mais leve do BERT criada a partir da técnica de destilação de conhecimento, do inglês *knowledge distillation* (SANH et al, 2019). Embora seja 40% menor, o DistilBERT é 60% mais rápido que o BERT e retém 97% de seu entendimento de linguagem, sendo ideal para computação on device (SANH et al., 2019).
- RoBERTa: Por sua vez, também em 2019, pesquisadores do Facebook AI apresentaram o RoBERTa (*Robustly Optimized BERT Pretraining Approach*), um modelo baseado no BERT que, além de ter sido treinado em um conjunto de dados maior do que o usado para o treinamento do BERT, usa a técnica de *dynamic masking* nesse processo (LIU et al., 2019). Essas melhorias permitiram que o RoBERTa alcançasse um desempenho superior ao BERT em várias tarefas de processamento de linguagem natural, incluindo a classificação de texto (LIU et al, 2019).
- XLNet: Ainda em 2019, pesquisadores do Google AI apresentaram o XLNet, um modelo de linguagem criado para suprir limitações identificadas pelos pesquisadores no BERT (YANG et al, 2019). Diferente do BERT, o XLNet o foi treinado na tarefa de modelagem de linguagem permutada, do inglês *Permutation Language Modeling*, que visa ajustar o modelo para a predição de palavras, assim como no preenchimento de lacunas, mas seguindo uma ordem aleatória, ao invés de realizar uma predição sequencial (PUROHIT, 2019). Esse modelo conseguiu superar o BERT no

cumprimento de 20 tarefas, dentre elas a análise de sentimentos e a resposta a perguntas.

- ALBERT: Também em 2019, pesquisadores da Google AI disponibilizaram o ALBERT (*A Lite BERT*). Assim como o DistilBERT, o ALBERT foi projetado para ser menor e mais leve do que o BERT a partir do uso das técnicas de *Factorized Embedding Parameterization* e *Cross-layer parameter sharing*, que visam a redução de parâmetros do modelo a partir do compartilhamento desses parâmetros entre as suas diversas camadas (LAN et al, 2019).
- Diversas outras soluções de NLP foram criadas a partir do ajuste fino dos modelos listados na seção anterior. Para este trabalho foram escolhidas soluções voltadas para a segurança de informação, como o cyBERT, por exemplo, uma versão do BERT pré-treinada para a detecção de anomalias em logs (RICHARDSON, 2019).
- Outros dois modelos avaliados neste trabalho foram, o SecBERT e o SecRoBERTa, versões do BERT e do RoBERTa, respectivamente, treinados em artigos do domínio de cibersegurança (SECBERT, 2020).

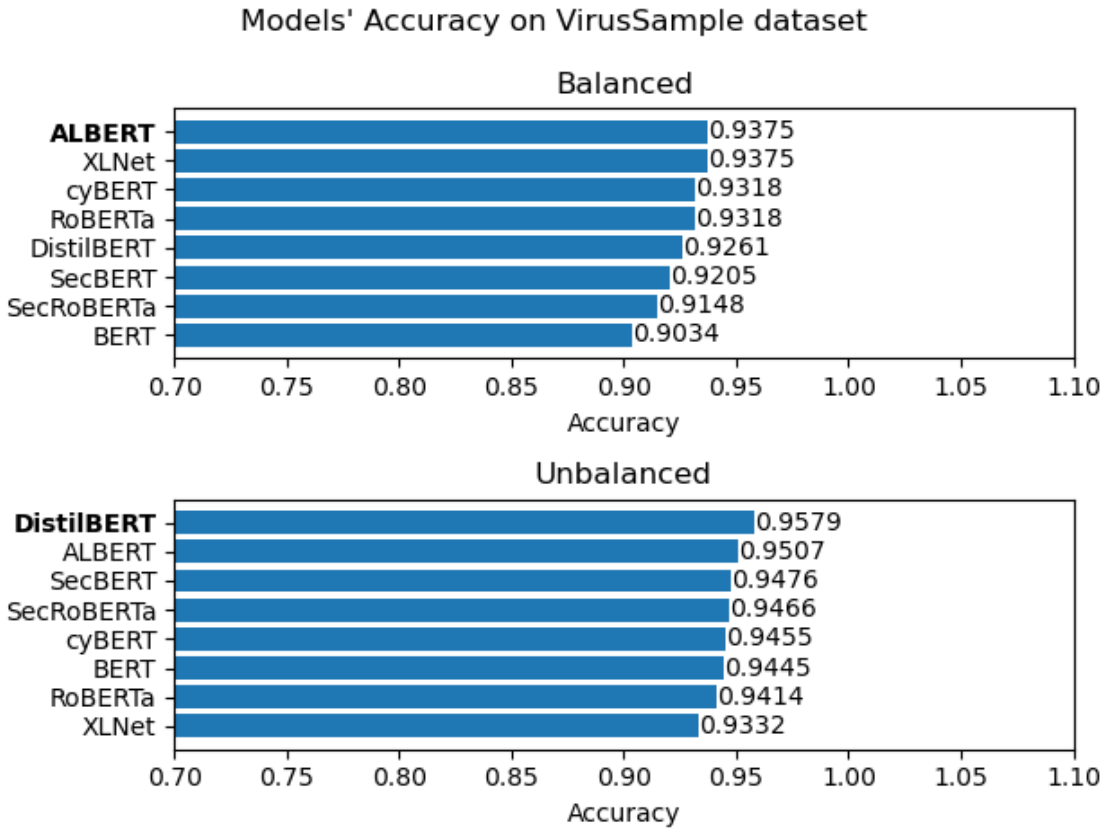
5 RESULTADOS

A escolha das métricas para comparação dos modelos foi feita com base nos experimentos de Düzgün et al. (2022), que usa o F1-Score e a Área sob a curva ROC como métricas comparativas entre os modelos estado da arte testados. Além dessas, a Acurácia e o Coeficiente de Correlação de Matthews também foram usados para comparação.

Este capítulo apresenta a comparação do desempenho dos modelos nas diferentes bases de dados com base nos experimentos descritos no capítulo anterior. Para melhor visualização dos resultados, gráficos gerados a partir dos valores obtidos para cada métrica em cada modelo serão apresentados adiante.

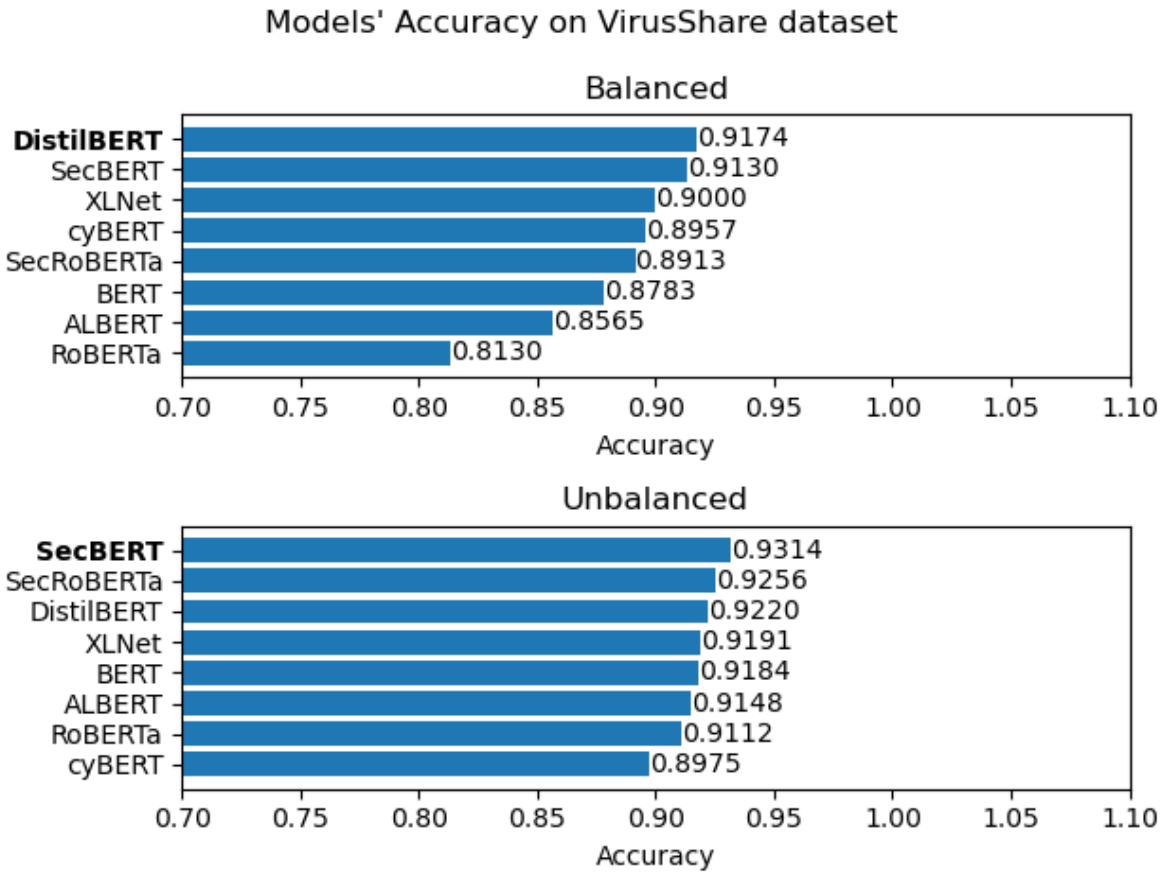
Os gráficos a seguir comparam o desempenho dos modelos com base em sua acurácia, métrica esta que visa medir a parcela de predições corretas feitas pelo modelo. Os gráficos em cada figura representam o desempenho dos modelos no dataset balanceado e no dataset não balanceado, respectivamente, estando os modelos ordenados do melhor ao pior desempenho.

Figura 4 - gráfico de acurácia dos modelos no dataset VirusSample.



Fonte: O autor (2023).

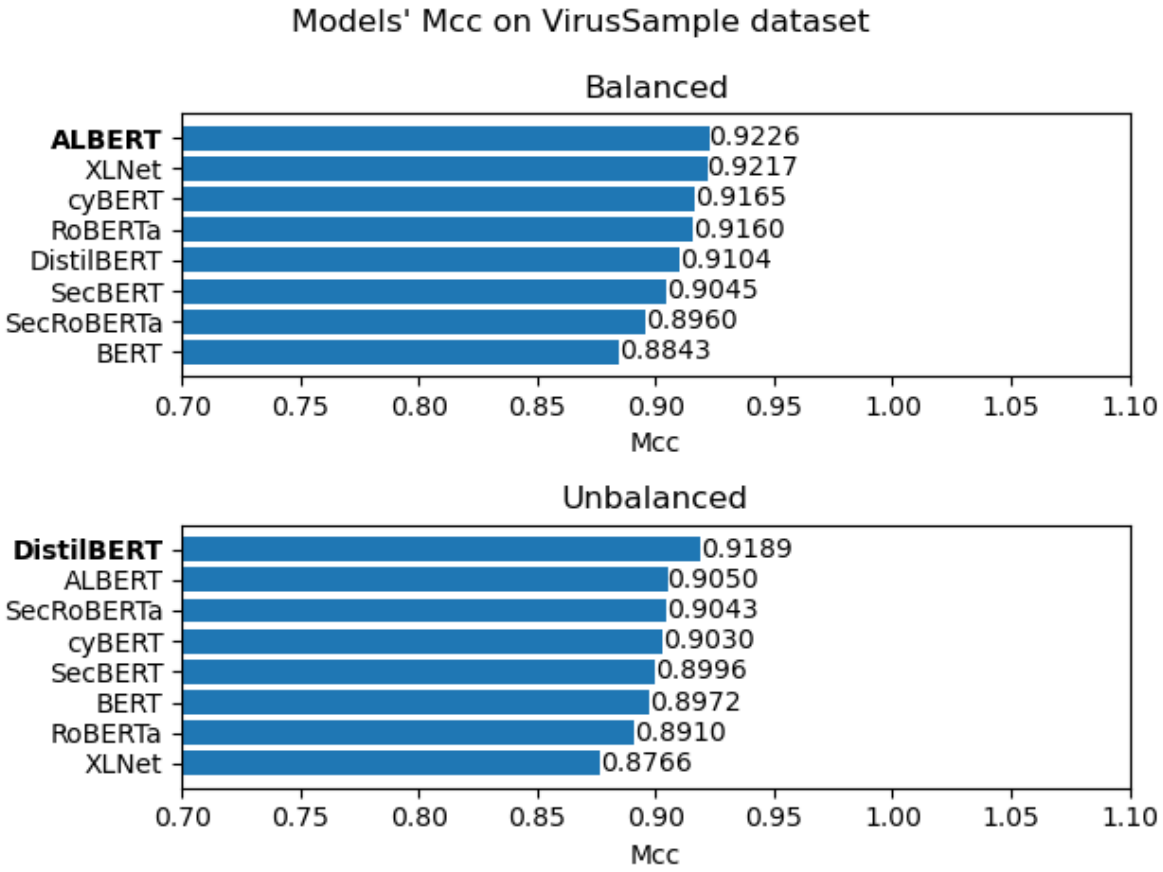
Figura 5 - gráfico de acurácia dos modelos no dataset VirusShare.



Fonte: O autor (2023).

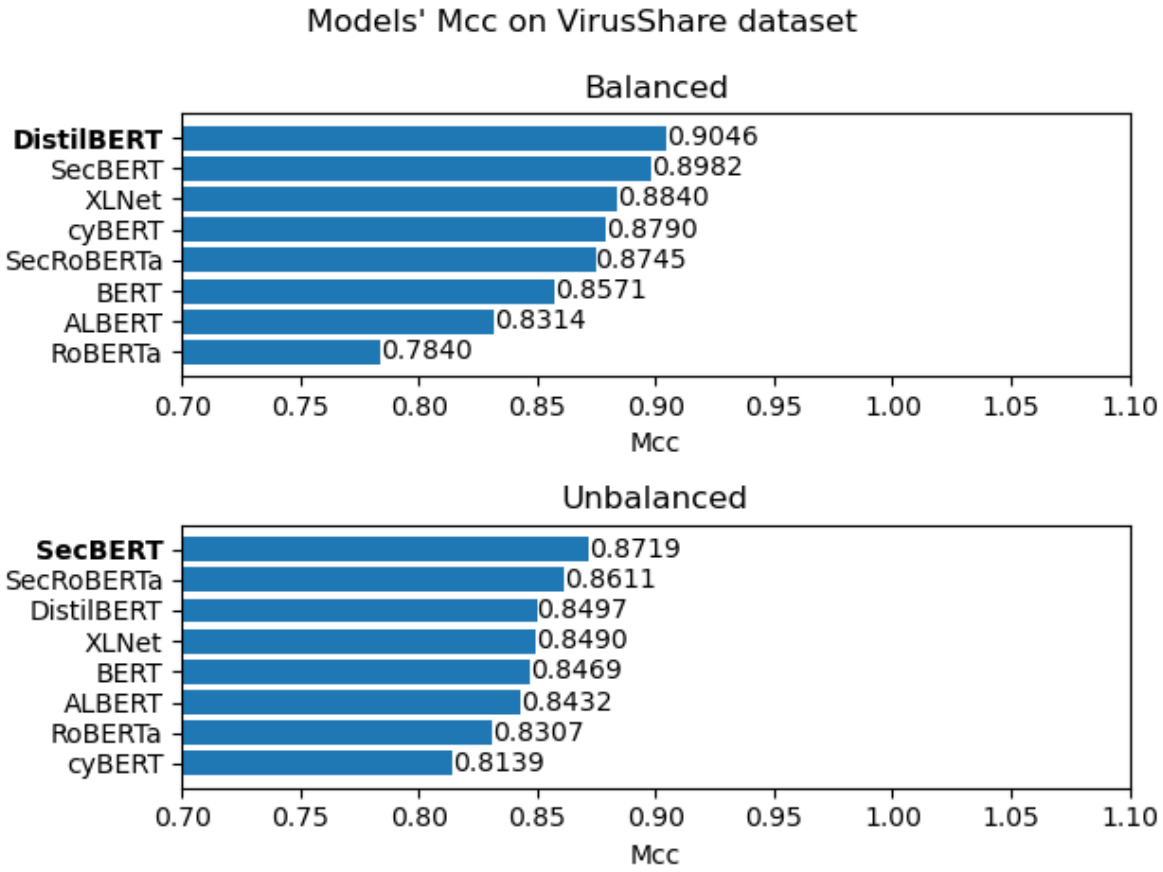
Os gráficos a seguir, por sua vez, comparam os modelos com base no Coeficiente de Correlação de Matthews, do inglês Matthews *Coefficient Correlation*, ou MCC, que visa medir a qualidade das predições feitas pelo modelo e produz resultados mais confiáveis, especialmente para bases sem balanceamento (CHICCO e JURMAN, 2020).

Figura 6 - gráfico de desempenho dos modelos no dataset VirusSample a partir do Coeficiente de Correlação de Matthews.



Fonte: O autor (2023).

Figura 7 - gráfico de desempenho dos modelos no dataset VirusShare a partir do Coeficiente de Correlação de Matthews.

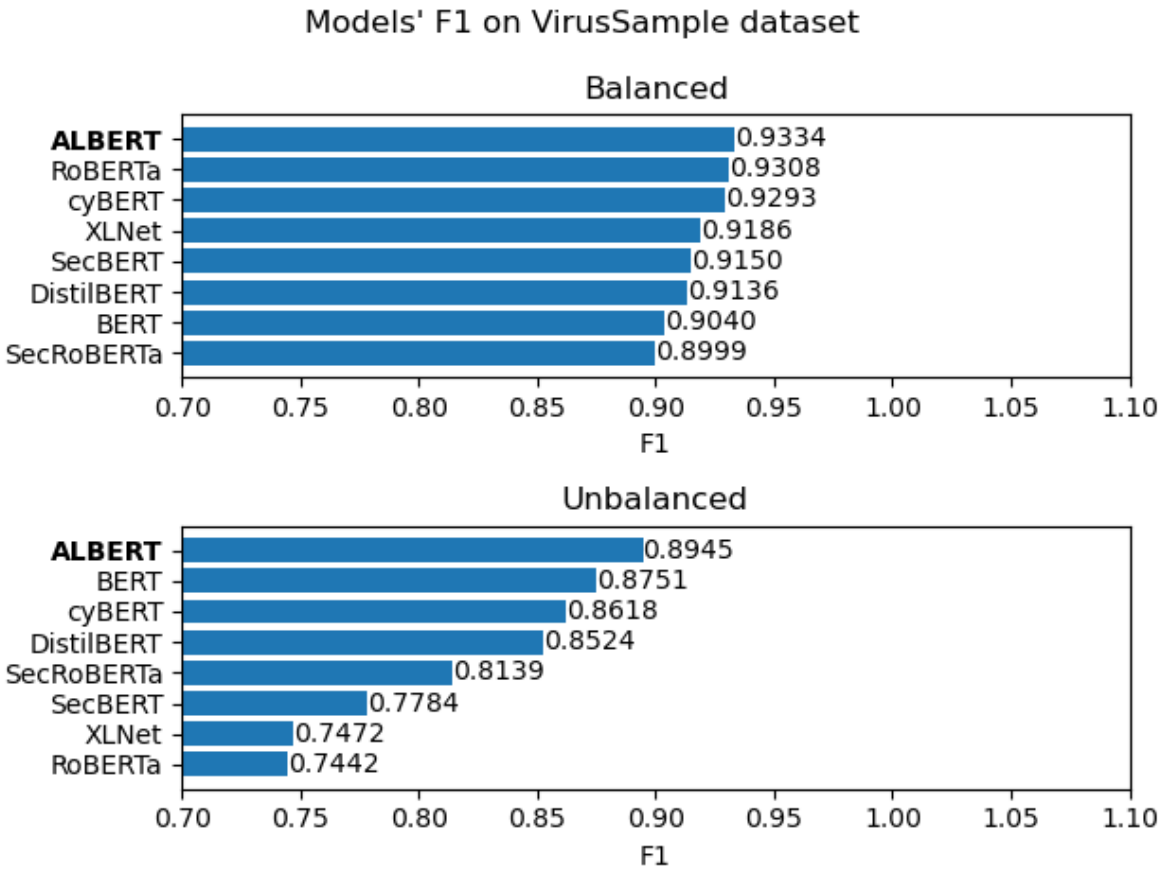


Fonte: O autor (2023).

Com base nas métricas acurácia e MCC, o ALBERT possui maior desempenho na versão balanceada da base de dados VirusSample, enquanto na versão original, o DistilBERT é o modelo com melhor desempenho. Por sua vez, o DistilBERT supera os demais modelos na versão balanceada da base VirusShare, enquanto o SecBERT se sai melhor que os demais na versão original dessa base.

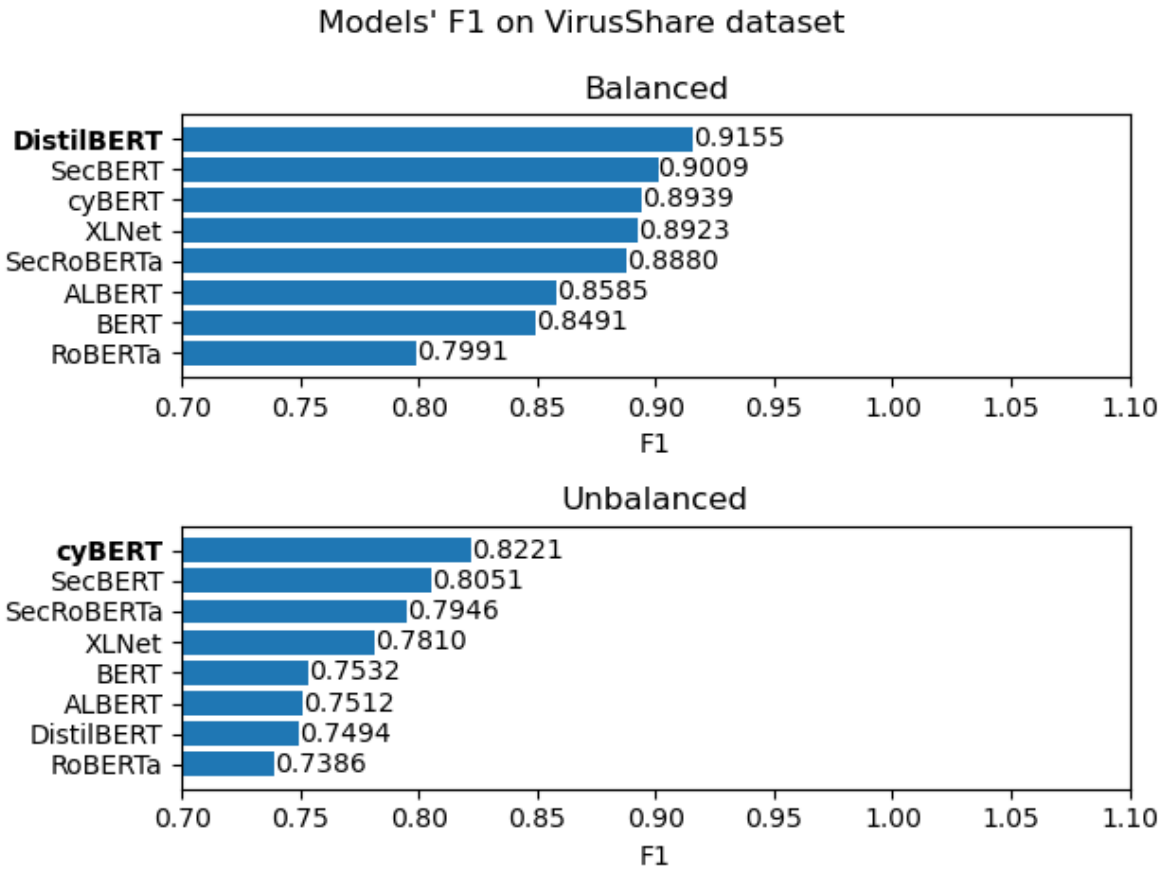
Já os próximos gráficos visam diferenciar os modelos com base na métrica F1-Score, que representa a média harmônica ponderada entre a precisão, medida que avalia a proporção de exemplos classificados corretamente como positivos em relação ao número total de exemplos classificados como positivos, e a evocação, do inglês *recall*, a proporção de exemplos positivos que foram classificados corretamente como positivos em relação ao número total de exemplos positivos reais.

Figura 8 - medida F1 dos modelos quando aplicados ao dataset VirusSample.



Fonte: O autor (2023).

Figura 9 - medida F1 dos modelos quando aplicados ao dataset VirusShare.

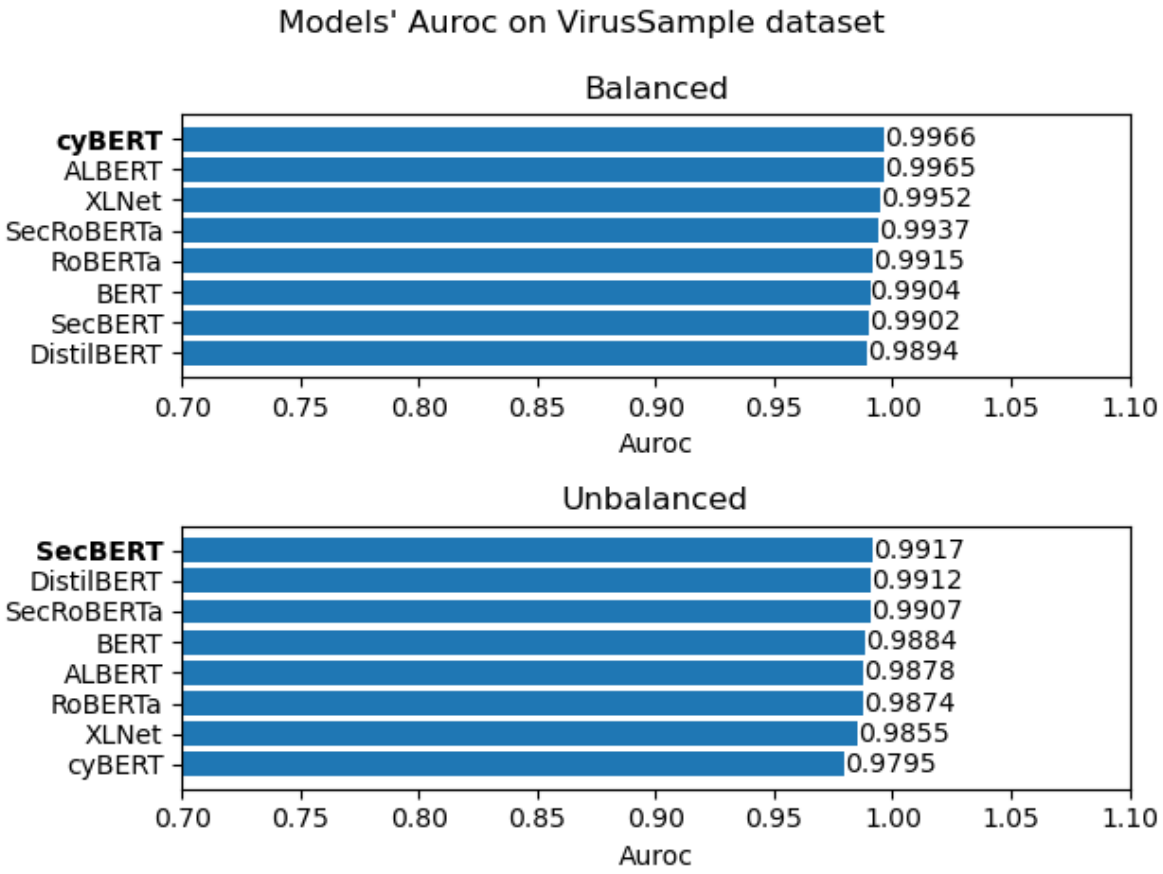


Fonte: O autor (2023).

Com relação a métrica F1, o modelo ALBERT possui o maior desempenho entre todos os modelos testados em ambas as versões da base de dados VirusSample. Já na base de dados VirusShare, o DistilBERT e o cyBERT foram os modelos com maior desempenho nas versões balanceada e original da base de dados, respectivamente.

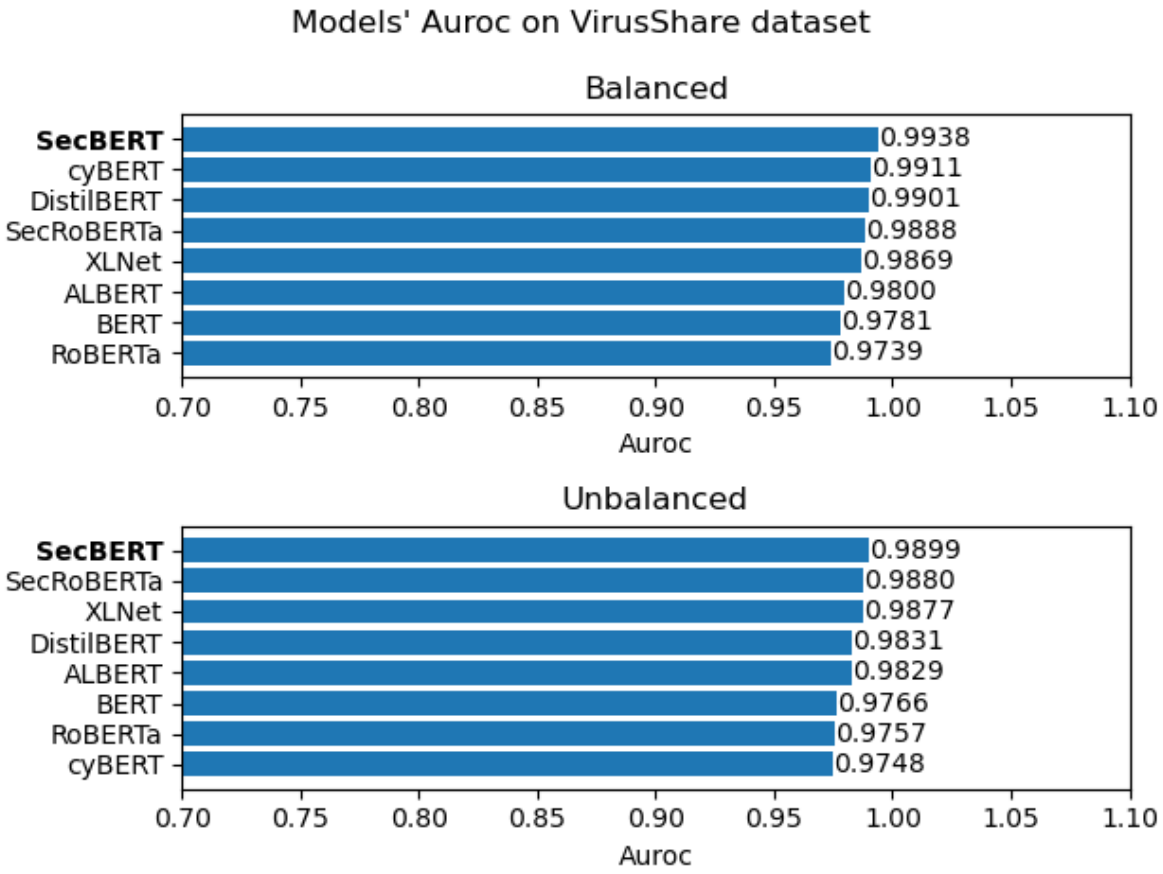
Por fim, os próximos gráficos comparam o desempenho dos modelos tendo como referência a métrica de Área sob a curva ROC, que visa medir o quão capaz os modelos são de distinguir entre as classes do dataset.

Figura 10 - medida de Área sob a curva ROC dos modelos avaliados no dataset VirusSample.



Fonte: O autor (2023).

Figura 11 - medida de Área sob a curva ROC dos modelos avaliados no dataset VirusShare.



Fonte: O autor (2023).

A partir da métrica de área sob a curva ROC, os modelos voltados ao domínio de segurança de informação alcançam maiores valores, com o SecBERT obtendo o melhor desempenhona versão não balanceada da base de dados VirusSample e em ambas as versões da base de dados VirusShare e com o cyBERT obtendo o melhor desempenho na versão balanceada da base de dados VirusSample.

As tabelas a seguir visam comparar os valores obtidos pelos modelos avaliados neste trabalho com os resultados dos modelos avaliados por Düzgün, et al. (2022). Os melhores valores obtidos em cada uma das métricas estão destacados em negrito.

Tabela 4 - Benchmark da base de dados VirusSample

	Original Version		Balanced Version	
Model	F1-Score	AUC Score	F1-Score	AUC Score
Düzgün, et al. (2022)				
Random Forest	0.55100	0.8962	0.8391	0.9688
SVM	0.7331	0.964	0.8975	0.99782
XGBoost	0.7351	**0.9816**	0.9031	**0.9941**
HGBoost	0.715	0.9776	0.8802	0.9841
LSTM	**0.7788**	0.9682	0.84	0.9478
BERT	0.7253	0.959	0.8948	0.9708
CANINE	0.7182	0.9621	**0.9086**	0.9828
Modelos avaliados neste trabalho				
ALBERT	**0.8945**	0.9878	**0.9334**	0.9965
BERT	0.8751	0.9884	0.904	0.9904
DistilBERT	0.8524	0.9912	0.9136	0.9894
RoBERTa	0.7442	0.9874	0.9308	0.9915
SecBERT	0.7784	**0.9917**	0.915	0.9902
SecRoBERTa	0.8139	0.9907	0.8999	0.9937
XLNet	0.7472	0.9855	0.9186	0.9952
cyBERT	0.8618	0.9795	0.9293	**0.9966**

Fonte: O autor a partir dos dados de Düzgün, et al. (2022).

Tabela 5 - Benchmark da base de dados VirusShare

	Original Version		Balanced Version	
Model	F1-Score	AUC Score	F1-Score	AUC Score
Düzgün, et al. (2022)				
Random Forest	0.60200	0.9334	0.6609	0.9304
SVM	**0.7343**	0.9226	**0.7581**	0.9404
XGBoost	0.7178	**0.9666**	0.8525	**0.9577**
HGBoost	0.6952	0.9582	0.747	0.9477
LSTM	0.7007	0.9359	0.7007	0.9359
BERT	0.7068	0.9432	0.7447	0.8843
CANINE	0.6955	0.9261	0.7284	0.9045
Modelos avaliados neste trabalho				
ALBERT	0.7512	0.9829	0.8585	0.98
BERT	0.7532	0.9766	0.8491	0.9781
DistilBERT	0.7494	0.9831	**0.9155**	0.9901
RoBERTa	0.7386	0.9757	0.7991	0.9739
SecBERT	0.8051	**0.9899**	0.9009	**0.9938**
SecRoBERTa	0.7946	0.988	0.888	0.9888
XLNet	0.781	0.9877	0.8923	0.9869
cyBERT	**0.8221**	0.9748	0.8939	0.9911

Fonte: O autor a partir dos dados de Düzgün, et al. (2022)

Modelos baseados em transformers, como ALBERT, BERT, cyBERT, DistilBERT, RoBERTa, SecBERT, SecRoBERTa e XLNet, apresentam desempenho geralmente superior aos métodos tradicionais, como Random Forest, SVM, XGBoost, HGBoost e LSTM, em termos de F1-Score e AUC Score, demonstrando sua eficácia na análise de malwares.

É importante notar que a versão balanceada do conjunto de dados tende a resultar em um desempenho melhor em comparação com a versão original, indicando que o balanceamento das classes pode ser crucial para aprimorar a precisão e a generalização dos modelos em tarefas de classificação.

6 CONCLUSÃO

Considerando os resultados obtidos, é possível concluir que tanto os modelos com menos parâmetros, como o ALBERT e DistilBERT, como os modelos maiores ajustados ao domínio de cibersegurança foram capazes de obter resultados superiores aos de modelos de detecção estática no estado da arte.

Entre os modelos de *transformers* avaliados, o DistilBERT e o ALBERT se destacam pelo tamanho reduzido de parâmetros e pelo bom desempenho na classificação. Isso sugere que esses modelos podem ser particularmente adequados para a análise estática e classificação de malwares, graças à sua combinação de desempenho e eficiência computacional.

Os modelos específicos para o domínio de cibersegurança, cyBERT, SecBERT e SecRoBERTa, também apresentam resultados promissores, ressaltando a importância de adaptar e otimizar modelos de linguagem para domínios específicos.

O desempenho do XLNet, apesar de ser um modelo permutado e treinado de forma diferente em comparação ao BERT, apresenta um F1-Score e AUC Score comparáveis aos de outros modelos de *transformers*. Isso indica que abordagens alternativas de treinamento, como a permutação, podem ser uma área interessante para futuras pesquisas na detecção e classificação de malwares.

A variação no desempenho entre os diferentes modelos de *transformers* sugere que a escolha da arquitetura e a configuração dos hiperparâmetros podem ter um impacto significativo na eficácia da classificação de malwares. Portanto, a experimentação e a otimização desses aspectos são cruciais para alcançar melhores resultados na prática.

Embora os modelos baseados em transformers apresentem um desempenho geralmente superior, é importante considerar *trade-offs* entre precisão, eficiência computacional e complexidade do modelo ao selecionar o método mais adequado para uma aplicação específica de análise de malwares.

7 TRABALHOS FUTUROS

Com base nos resultados obtidos neste trabalho, trabalhos futuros podem focar na classificação usando uma base de dados binária, isto é, contendo amostras de arquivos maliciosos e não maliciosos. Isto permitirá a avaliação dos modelos como software antivírus. Outra possível abordagem é o ajuste fino de diferentes arquiteturas em documentos do domínio de cibersegurança voltados especificamente ao campo da detecção de malwares.

Existem diversas dificuldades para a detecção estática de malware, como a constante evolução dos modelos e o surgimento de novas técnicas de ofuscação, além dos já conhecidos softwares de ofuscação executáveis, como *packers* e *crypters* (ABOAOJA et al, 2022). Uma vez que as bases de dados são compostas por amostras de malware de um período de tempo específico, é possível que amostras mais recentes apresentem características diferentes daquelas usadas pelas bases de dados adotadas neste trabalho.

REFERÊNCIAS

ALAMMAR, Jay. **The Illustrated BERT, ELMo, and co. (How NLP Cracked Transfer Learning)**. 2018. Disponível em: https://jalammar.github.io/illustrated-bert/. Acesso em: 11 fev. 2023.

CHOPRA, Abhimanyu; PRASHAR, Abhinav; SAIN, Chandresh. Natural Language Processing. **International Journal Of Technology Enhancements And Emerging Engineering Research**. [S.l.], p. 131-134. nov. 2013. Disponível em: https://paper.researchbib.com/view/issn/2347-4289/1/4. Acesso em: 12 fev. 2023.

KASPERSKY. **Kaspersky descobre 380 mil arquivos maliciosos por dia em 2021**. 2022. Disponível em: https://www.kaspersky.com.br/about/press-releases/2022_kaspersky-descobre-380-mil-arquivos-maliciosos-por-dia-em-2021. Acesso em: 14 mar. 2022.

ASLAN, Ömer Aslan; SAMET, Refik. A Comprehensive Review on Malware Detection Approaches. **Ieee Access**, [S.l.], v. 8, n. 1, p. 6249-6271, 3 jan. 2020. Disponível em: https://ieeexplore.ieee.org/document/8949524. Acesso em: 14 mar. 2022.

GRÉGIO, André Ricardo Abed. **Malware Behavior**: comportamento de programas maliciosos. 2012. 156 f. Tese (Doutorado) - Curso de Engenharia Elétrica, Universidade Federal de Campinas, Campinas, 2012. Disponível em: https://www.lasca.ic.unicamp.br/paulo/teses/20121128-PhD-Andre.Ricardo.Abed.Gregio-Malware.behavior.pdf. Acesso em: 15 mar. 2022.

TURING, Alan Mathison. Computing Machinery and Intelligence. **Mind**. Oxford, p. 433-460. 01 out. 1950. Disponível em: https://academic.oup.com/mind/article/LIX/236/433/986238. Acesso em: 12 fev. 2023.

IBM. Ibm. **Machine Learning.** 2020. Disponível em: https://www.ibm.com/cloud/learn/machine-learning. Acesso em: 15 mar. 2022.

HUTCHINS, John. **The first public demonstration of machine translation**: the Georgetown-IBM system, 7th January 1954. [S.l.]: [S.N.]., 2005. Disponível em: https://hutchinsweb.me.uk/GU-IBM-2005.pdf. Acesso em: 12 fev. 2023.

SAMUEL, Arthur L.. Some Studies in Machine Learning Using the Game of Checkers. **Ibm Journal Of Research And Development.** [S.l.], p. 210-229. jul. 1959. Disponível em: https://hci.iwr.uni-heidelberg.de/system/files/private/downloads/636026949/report_frank_gabel.pdf. Acesso em: 15 mar. 2022.

BRIGUGLIO, William; ELMILIGI, Haytham; SAAD, Sherif. The Curious Case of Machine Learning In Malware Detection. In: INTERNATIONAL CONFERENCE ON INFORMATION SYSTEMS SECURITY AND PRIVACY, 5., 2019, Praga. **Proceedings of the 5th International Conference on Information Systems Security and Privacy.** Praga: Scitepress, 2019. p. 528-535. Disponível em: https://arxiv.org/abs/1905.07573v1. Acesso em: 15 mar. 2022.

CARVALHO, Marco M.; CHAN, Philip K.; HASSEN, Mehadi. Malware classification using static analysis based features. In: IEEE SYMPOSIUM SERIES ON COMPUTATIONAL INTELLIGENCE (SSCI), 1., 2017, [S.l.]. **Proceedings of the 2017 IEEE Symposium Series on Computational Intelligence (SSCI).** Honolulu: Ieee, 2017. p. 1-7. Disponível em: https://ieeexplore.ieee.org/document/8285426. Acesso em: 15 mar. 2022.

CAVAZOS, John; LA ROSA, Leonardo de; KILGALLON, Sean. Improving the effectiveness and efficiency of dynamic malware analysis with machine learning. In: RESILIENCE WEEK, 1., 2017, Wilmington. **Proceedings of the 2017 Resilience Week (RWS).** Wilmington: Ieee, 2017. p. 30-36. Disponível em: https://ieeexplore.ieee.org/document/8088644. Acesso em: 15 mar. 2022.

NEELAMEGAM, S.; RAMARAJ, E.. Classification algorithm in Data mining: an overview. **International Journal Of P2P Network Trends And Technology (Ijptt)**. Baghdad, p. 369-374. 5 set. 2013. Disponível em: https://www.ijpttjournal.org/archives/ijptt-v3i8p101. Acesso em: 15 ago. 2022.

GAMA, J.; BRAZDIL, P. **Characterization of classification algorithms**. In: PINTO FERREIRA, C.; MAMEDE, N. J. (Ed.). Progress in Artificial Intelligence. Berlin, Heidelberg: Springer Berlin Heidelberg, 1995. p. 189–200. ISBN 978-3-540-45595-0.

ALMEIDA, Felipe; XEXÉO, Geraldo. **Word Embeddings**: a survey. CoRR. [S.l.], n. p. 25 jan. 2019. Disponível em: https://arxiv.org/abs/1901.09069v1. Acesso em: 04 set. 2021.

SONICWALL. **Mid-Year Update Cyber Threat Report**. 2022. Disponível em: https://www.sonicwall.com/medialibrary/en/white-paper/mid-year-2022-cyber-threat-report.pdf. Acesso em: 17 jan. 2023.

CHANG, Ming-Wei; DEVLIN, Jacob; LEE, Kenton; TOUTANOVA, Kristina. BERT: pre-training of deep bidirectional transformers for language. **Corr**. [S.l.], p. n.p. 30 out. 2018. Disponível em: https://arxiv.org/abs/1810.04805v2. Acesso em: 05 jan. 2023.

WINOGRAD, Terry. **Procedures as a Representation for Data in a Computer Program for Understanding Natural Language**. 1971. 461 f. Tese (Doutorado) - Curso de Philosophy, Massachusetts Institute Of Technology, [S.l.], 1971. Disponível em: http://hci.stanford.edu/winograd/shrdlu/AITR-235.pdf. Acesso em: 13 fev. 2023.

WEIZENBAUM, Joseph. ELIZA—a computer program for the study of natural language communication between man and machine. **Communications Of The Acm**, New York, v. 9, n. 1, p. 36-45, jan. 1966. Disponível em: https://dl.acm.org/doi/10.1145/365153.365168. Acesso em: 13 fev. 2023.

BROWN, P. et al. A Statistical Approach to Language Translation. In: CONFERENCE ON COMPUTATIONAL LINGUISTICS, 12., 1988, Yorktown Heights. **COLING '88: Proceedings of the 12th conference on Computational linguistics**. [S.l.]: Association For Computational Linguistics, 1988. p. 71-76. Disponível em: https://dl.acm.org/doi/10.3115/991635.991651. Acesso em: 13 fev. 2023.

NARASIMHAN, Karthik et al.. **Openai**. Improving Language Understanding by Generative Pre-Training. 2018. Disponível em: https://cdn.openai.com/research-covers/language-unsupervised/language_understanding_paper.pdf. Acesso em: 14 fev. 2023.

TURNEY, Peter. Thumbs Up or Thumbs Down? Semantic Orientation Applied to Unsupervised Classification of Reviews. In: ANNUAL MEETING OF THE ASSOCIATION FOR COMPUTATIONAL LINGUISTICS, 40., 2002, Philadelphia. **Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics**. [S.l.]: [S.N.], 2002. p. 417-424. Disponível em: https://arxiv.org/abs/cs/0212032. Acesso em: 14 fev. 2023.

FREITAG, Dayne. Machine Learning for Information Extraction in Informal Domains. **Machine Learning**. Pittsburgh, p. 169-202. jan. 2000. Disponível em: http://www.cs.bilkent.edu.tr/~guvenir/courses/CS550/Seminar/freitag2000-ml.pdf. Acesso em: 14 fev. 2023.

SINCLAIR, Neil. A Gentle Introduction to Transfer Learning in NLP. **Towards Data Science**, [S.l.], 27 jan. 2021. Disponível em: https://chat.openai.com/chat/ad4ebe3e-4ebf-45c8-8496-f8cbc3a1f5fb. Acesso em: 15 fev. 2023.

VASWANI, Ashish et al. Attention Is All You Need. In: CONFERENCE ON NEURAL INFORMATION PROCESSING SYSTEMS, 30., 2017, Long Beach. **Advances in**

neural information processing systems. Long Beach: [S.N.], 2017. p. 5998-6008. Disponível em: https://proceedings.neurips.cc/paper/2017/hash/3f5ee243547dee91fbd053c1c4a845aa-Abstract.html. Acesso em: 15 fev. 2023.

ALAMMAR, Jay. **The Illustrated Transformer**. 2018. Disponível em: https://jalammar.github.io/illustrated-transformer/. Acesso em: 15 fev. 2023.

MCCULLOCH, Warren; PITTS, Walter. A logical calculus of the ideas immanent in nervous activity. **Bulletin Of Mathematical Biophysics**, [S.l.], v. 5, n. 1, p. 115-133, dez. 1943. Disponível em: https://link.springer.com/article/10.1007/bf02478259. Acesso em: 15 fev. 2023.

ROSENBLATT, Frank. The Perceptron: a probabilistic model for information storage and organization in the brain. **Psychological Review**. [S.l.], p. 386-408. nov. 1958.

MINSKY, Marvin; PAPERT, Seymour. **Perceptrons**: an introduction to computational geometry. [S.l.]: Mit Press, 1969.

RAO, C. Radhakrishna; MITRA, Sujit Kumar. Generalized inverse of a matrix and its applications. In: **Proceedings of the Sixth Berkeley Symposium on Mathematical Statistics and Probability, Volume 1: Theory of Statistics**. University of California Press, 1972. p. 601-621.

COVER, Thomas; HART, Peter. Nearest neighbor pattern classification. **IEEE transactions on information theory**, v. 13, n. 1, p. 21-27, 1967.

BERKSON, Joseph. Application of the logistic function to bio-assay. **Journal of the American statistical association**, v. 39, n. 227, p. 357-365, 1944.

FISHER, Ronald A. The use of multiple measurements in taxonomic problems. **Annals of eugenics**, v. 7, n. 2, p. 179-188, 1936.

GOOD, Isidore Jacob. Probability and the Weighing of Evidence. **Philosophy**, v. 26, n. 97, 1950.

RUMELHART, David E.; HINTON, Geoffrey E.; WILLIAMS, Ronald J. Learning representations by back-propagating errors. **Nature**, v. 323, n. 6088, p. 533-536, 1986.

TAHIR, Rabia. A Study on Malware and Malware Detection Techniques. **International Journal Of Education And Management Engineering**. [S.l.], p. 20-30. 8 mar. 2018. Disponível em: https://www.mecs-press.org/ijeme/. Acesso em: 19 fev. 2023.

GILL, Harjeevan. Malware: Types, Analysis and Classifications. [S. l.]: **Open Engineering Inc**, 21 jun. 2022. DOI 10.31224/2423. Disponível em: http://dx.doi.org/10.31224/2423. Acesso em 19 fev. 2023.

TROJAN dropper. **Malwarebytes Labs**. Cork. 30 jan. 2020. Disponível em: https://www.malwarebytes.com/blog/threats/trojan-dropper. Acesso em: 19 fev. 2023.

AL-HAWAWREH, Muna; HARTOG, Frank Den; SITNIKOVA, Elena. Targeted Ransomware: a new cyber threat to edge system of brownfield industrial internet of things. **IEEE Internet Of Things Journal**. New York, p. 7137-7151. maio 2019. Disponível em: https://ieeexplore.ieee.org/document/8703829. Acesso em: 19 fev. 2023.

MELTZER, Tom; PHILLIPS, Sarah. From the first email to the first YouTube video: a definitive internet history. **The Guardian**. Londres. 23 de out. de 2009. Disponível em: <https://www.theguardian.com/technology/2009/oct/23/internet-history>. Acesso em: 22 de jan. de 2023.

CHEN, Thomas M.; ROBERT, Jean-Marc. The evolution of viruses and worms. In: **Statistical methods in computer security**. CRC press, 2004. p. 289-310. Disponível em: <https://web.archive.org/web/20090517083356/http://vx.netlux.org/lib/atc01.html>. Acesso em: 22 fev. 2023

KORET, Joxean; BACHAALANY, Elias. **The Antivirus Hacker's Handbook**. Indianapolis: John Wiley & Sons, 2015. 359 p.

G DATA SOFTWARE. **Company Profile**. 2017. Disponível em: https://web.archive.org/web/20170315111115/https://www.gdatasoftware.com/about-g-data/company-profile. Acesso em: 22 fev. 2023.

JACKSON, Keith. Dr Solomon's anti-virus toolkit. **Computer Fraud & Security Bulletin.** [S.l.], p. 9-13. ago. 1989. Disponível em: https://doi.org/10.1016/0142-0496(89)90163-X. Acesso em: 22 fev. 2023.

CHAKKARAVARTHY, S. Sibi; SANGEETHA, D.; VAIDEHI, V.. A Survey on malware analysis and mitigation techniques. **Computer Science Review.** [S.l.], p. 1-23. maio 2019. Disponível em: https://doi.org/10.1016/j.cosrev.2019.01.002. Acesso em: 22 fev. 2023.

GANDOTRA, Ekta; BANSAL, Divya; SOFAT, Sanjeev. Malware analysis and classification: A survey. **Journal of Information Security**, v. 2014, mar. 2014. Disponível em: https://www.scirp.org/html/4-7800194_44440.htm. Acesso em: 26 fev. 2023.

ARIFFIN, Khairul Akram Zainol; OMAR, Khairuddin; SIHWAIL, Rami. A Survey on Malware Analysis Techniques: static, dynamic, hybrid and memory analysis. **International Journal On Advanced Science, Engineering And Information Technology.** [S.l.], p. 1662-1671. 2018. Disponível em: http://ijaseit.insightsociety.org/index.php?option=com_content&view=article&id=9&Itemid=1&article_id=6827. Acesso em: 26 fev. 2023.

INTERNATIONAL INFORMATION SYSTEM SECURITY CERTIFICATION CONSORTIUM. **Cybersecurity Workforce Study**. [S.l.]: International Information System Security Certification Consortium, 2022. 86 p. Disponível em: https://www.isc2.org/Research/Workforce-Study. Acesso em: 26 fev. 2023.

COULTER, Martin. Cybersecurity insiders say big companies use NDAs to hide data breaches, potentially avoiding millions of dollars in fines. **Business Insider**. [S.l.], 2019. Disponível em: https://www.businessinsider.com/cybersecurity-insiders-reveal-nda-hide-data-breach-2019. Acesso em: 26 fev. 2023.

MORGAN, Steve. Top 10 Cybersecurity Predictions And Statistics For 2023. **Cybercrime Magazine**. Northport, 10 dez. 2022. Disponível em: https://cybersecurityventures.com/top-5-cybersecurity-facts-figures-predictions-and-statistics-for-2021-to-2025/. Acesso em 26 fev. 2023.

DÜZGÜN, Berkant et al. Benchmark Static API Call Datasets for Malware Family Classification. In: **2022 7th International Conference on Computer Science and Engineering (UBMK)**. IEEE, 2022. p. 1-5. Disponível em: https://ieeexplore.ieee.org/abstract/document/9919580. Acesso em 13 mar. 2023.

RAHALI, Abir; AKHLOUFI, Moulay A. MalBERT: Malware Detection using Bidirectional Encoder Representations from Transformers. In: **2021 IEEE International Conference on Systems, Man, and Cybernetics (SMC)**. IEEE, 2021. p. 3226-3231. Disponível em: https://ieeexplore.ieee.org/abstract/document/9659287. Acesso em: 14 mar. 2023.

LIU, Yinhan et al. Roberta: A robustly optimized bert pretraining approach. **arXiv preprint arXiv:1907.11692**, 2019. Disponível em: https://arxiv.org/pdf/1907.11692.pdf. Acesso em: 21 mar. 2023.

YANG, Zhilin et al. Xlnet: Generalized autoregressive pretraining for language understanding. **Advances in neural information processing systems**, v. 32, 2019. Disponível em: https://proceedings.neurips.cc/paper/2019/hash/dc6a7e655d7e5840e66733e9ee67cc69-Abstract.html. Acesso em 27 de mar. 2023.

PUROHIT, Karan. Understand how the XLNet outperforms BERT in Language Modelling. **Medium**, 10 jul. 2019. Disponível em: https://medium.com/saarthi-ai/xlnet-the-permutation-language-model-b30f5b4e3c1 Acesso em: 27 mar. 2023.

RICHARDSON, Bartley. cyBERT. **Medium**, 05 dez. 2019. Disponível em: https://medium.com/rapids-ai/cybert-28b35a4c81c4. Acesso em: 28 mar. 2023.

LAN, Zhenzhong et al. Albert: A lite bert for self-supervised learning of language representations. **arXiv preprint arXiv:1909.11942**, 2019. Disponível em: https://arxiv.org/pdf/1909.11942.pdf. Acesso em: 28 mar. 2023.

SECBERT: A Pretrained Language Model for Cyber Security. A Pretrained Language Model for Cyber Security. **Hugging Face**. 2020. Disponível em: https://huggingface.co/jackaduma/SecBERT. Acesso em: 29 mar. 2023.

CHICCO, Davide; JURMAN, Giuseppe. The advantages of the Matthews correlation coefficient (MCC) over F1 score and accuracy in binary classification evaluation. **BMC genomics**, v. 21, p. 1-13, 2020. Disponível em: https://link.springer.com/article/10.1186/s12864-019-6413-7. Acesso em: 04 de abr. 2023.

PÉREZ-SÁNCHEZ, Antonio; PALACIOS, Rafael. Evaluation of local security event management system vs. standard antivirus software. **Applied Sciences**, v. 12, n. 3, p. 1076, 2022. Disponível em: https://www.mdpi.com/2076-3417/12/3/1076. Acesso em: 06 abr. 2023.

ABOAOJA, Faitouri A. et al. Malware detection issues, challenges, and future directions: A survey. **Applied Sciences**, v. 12, n. 17, p. 8482, 2022. Disponível em: https://www.mdpi.com/2076-3417/12/17/8482. Acesso em: 06 abr. 2023

YANG, Xiaofei et al. Hyperspectral image transformer classification networks. **IEEE Transactions on Geoscience and Remote Sensing**, v. 60, p. 1-15, 2022. Disponível em: https://www.mdpi.com/2076-3417/12/17/8482. Acesso em: 06 abr. 2023.

CHEN, Ke et al. HTS-AT: A hierarchical token-semantic audio transformer for sound classification and detection. In: **ICASSP 2022-2022 IEEE International Conference on**

Acoustics, Speech and Signal Processing (ICASSP). IEEE, 2022. p. 646-650. Disponível em: https://ieeexplore.ieee.org/abstract/document/9746312. Acesso em: 06 abr. 2023.

HENDRYCKS, Dan; LEE, Kimin; MAZEIKA, Mantas. Using pre-training can improve model robustness and uncertainty. In: **International Conference on Machine Learning**. PMLR, 2019. p. 2712-2721. Disponível em: https://proceedings.mlr.press/v97/hendrycks19a.html. Acesso em 06 abr. 2023.

GIBERT, Daniel; MATEU, Carles; PLANES, Jordi. The rise of machine learning for detection and classification of malware: Research developments, trends and challenges. **Journal of Network and Computer Applications**, v. 153, p. 102526, 2020. Disponível em: https://www.sciencedirect.com/science/article/pii/S1084804519303868. Acesso em 06 abr. 2023.

RAD, Babak Bashari; MASROM, Maslin; IBRAHIM, Suhaimi. Evolution of computer virus concealment and anti-virus techniques: a short survey. **arXiv preprint arXiv:1104.1070**, 2011. Disponível em: https://arxiv.org/abs/1104.1070. Acesso em: 06 abr. 2023.

GUINIER, Daniel. Computer "virus" identification by neural networks: An artificial intelligence connectionist implementation naturally made to work with fuzzy information. **ACM SIGSAC Review**, v. 9, n. 4, p. 49-59, 1991. Disponível em: https://dl.acm.org/doi/abs/10.1145/126569.127021. Acesso em: 06 abr. 2023.

SENEVIRATNE, Sachith et al. Self-Supervised Vision Transformers for Malware Detection. **IEEE Access**, v. 10, p. 103121-103135, 2022. Disponível em: https://ieeexplore.ieee.org/abstract/document/9889730. Acesso em: 06 abr. 2023.

SHIVANI. Future of Antivirus Protection: Top Antivirus Software Using AI. **IndustryWired**. [S.l], 29 set. 2021. Disponível em: https://industrywired.com/future-of-antivirus-protection-top-antivirus-software-using-ai/. Acesso em: 06 abr. 2023.

TORREY, Lisa; SHAVLIK, Jude. Transfer learning. In: **Handbook of research on machine learning applications and trends: algorithms, methods, and techniques**. IGI global, 2010. p. 242-264. Disponível em: https://www.igi-global.com/chapter/transfer-learning/36988. Acesso em: 21 abr. 2023.

AZEEZ, Nureni Ayofe et al. Windows PE malware detection using ensemble learning. In: **Informatics**. MDPI, 2021. p. 10. Disponível em: https://www.mdpi.com/2227-9709/8/1/10 Acesso em: 23 abr. 2023.

HUANG, Xiang et al. A method for windows malware detection based on deep learning. **Journal of Signal Processing Systems**, v. 93, p. 265-273, 2021. Disponível em: https://link.springer.com/article/10.1007/s11265-020-01588-1. Acesso em 23 abr. 2023.

DEMETRIO, Luca et al. Adversarial exemples: A survey and experimental evaluation of practical attacks on machine learning for windows malware detection. **ACM Transactions on Privacy and Security (TOPS)**, v. 24, n. 4, p. 1-31, 2021. Disponível em: https://dl.acm.org/doi/pdf/10.1145/3473039. Acesso em: 23 abr. 2023.

ALSMADI, Tibra; ALQUDAH, Nour. A Survey on malware detection techniques. In: **2021 International Conference on Information Technology (ICIT)**. IEEE, 2021. p. 371-376. Disponível em: https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=9491765. Acesso em: 23 abr. 2023.

ASSEGIE, Tsehay Admassu. An optimized KNN model for signature-based malware detection. **International Journal of Computer Engineering In Research Trends (IJCERT)**, ISSN, p. 2349-7084, 2021. Disponível em: https://papers.ssrn.com/sol3/papers.cfm?abstract_id=3814215. Acesso em: 23. abr 2023.

NIKOLOPOULOS, Stavros D.; POLENAKIS, Iosif. A graph-based model for malware detection and classification using system-call groups. **Journal of Computer Virology and Hacking Techniques**, v. 13, n. 1, p. 29-46, 2017. Disponível em: https://link.springer.com/article/10.1007/s11416-016-0267-1. Acesso em: 23 abr. 2023

JARAMILLO, Luis Eduardo Suástegui. Malware detection and mitigation techniques: Lessons learned from Mirai DDOS attack. **Journal of Information Systems Engineering & Management**, v. 3, n. 3, p. 19, 2018. Disponível em: https://www.jisem-journal.com/download/malware-detection-and-mitigation-techniques-lessons-learned-from-mirai-ddos-attack.pdf. Acesso em 23 abr. 2023

YE, Yanfang et al. DeepAM: a heterogeneous deep learning framework for intelligent malware detection. **Knowledge and Information Systems**, v. 54, p. 265-285, 2018. Disponível em: https://link.springer.com/article/10.1007/s10115-017-1058-9. Acesso em: 23 abr. 2023

GALAL, Hisham Shehata; MAHDY, Yousef Bassyouni; ATIEA, Mohammed Ali. Behavior-based features model for malware detection. **Journal of Computer Virology and Hacking Techniques**, v. 12, p. 59-67, 2016. Disponível em: https://link.springer.com/article/10.1007/s11416-015-0244-0. Acesso em: 23 abr. 2023.

SANTOS, Igor et al. Opcode sequences as representation of executables for data-mining-based unknown malware detection. **Information Sciences**, v. 231, p. 64-82, 2013. Disponível em: https://www.sciencedirect.com/science/article/abs/pii/S0020025511004336. Acesso em: 23. abr. 2023.