Instituto Federal Catarinense
Curso de Bacharelado em Ciência da Computação
Campus Blumenau

GUSTAVO LOFRESE CARVALHO

MODELOS DE LINGUAGEM COMO VETORES PARA GERAÇÃO DE CÓDIGOS MALICIOSOS: UM ESTUDO SOBRE DETECTABILIDADE E BARREIRAS DE SEGURANÇA

Blumenau
2025

GUSTAVO LOFRESE CARVALHO

MODELOS DE LINGUAGEM COMO VETORES PARA GERAÇÃO DE CÓDIGOS MALICIOSOS: UM ESTUDO SOBRE DETECTABILIDADE E BARREIRAS DE SEGURANÇA

Pré-Projeto do Trabalho de Conclusão de Curso apresentado ao Instituto Federal Catarinense — Campus Blumenau, como requisito parcial para obtenção do grau de Bacharel em Ciência da Computação.
Orientador: Ricardo de la Rocha Ladeira, Me.
Coorientador: Gabriel Eduardo Lima, Bacharel

Blumenau
2025

SUMÁRIO

1 INTRODUÇÃO

O advento do ChatGPT[1], em novembro de 2022, foi um marco no campo da Inteligência Artificial (IA), tornando-se rapidamente uma das tecnologias mais populares no mundo inteiro. Nos primeiros dois meses, a ferramenta gerida pelo grupo OpenAI alcançou mais de 100 milhões de usuários, evidenciando um grande impacto na área dos Modelos de Linguagem de Grande Escala (*Large Language Models - LLMs*) (GUPTA, M. et al., 2023). Os LLMs utilizam técnicas de Processamento de Linguagem Natural (Natural Language Processing - NLP) para reconhecer padrões linguísticos em dados e gerar respostas a comandos. (MADANI, 2023; YAMIN; HASHMI; KATT, 2024).

Esses modelos são altamente versáteis, sendo capazes de processar textos, sons e imagens (GUPTA, M. et al., 2023). No entanto, a principal interação com os LLMs ocorre por meio de *prompts* textuais, que são instruções fornecidas como forma de entrada para orientar a geração de respostas alinhadas às solicitações específicas (AL-KARAKI; KHAN; OMAR, 2024). Um dos principais usos dessa interação é a geração de código, na qual os LLMs produzem trechos de programas e *softwares* a partir dos padrões aprendidos, mesmo sem possuírem um entendimento real de programação (MADANI, 2023). Essa característica acaba por gerar implicações diretas no ramo da cibersegurança, uma vez que a popularização das IAs afeta diretamente o âmbito dos ataques digitais, tanto defensivamente quanto ofensivamente (GUPTA, M. et al., 2023).

Uma das principais consequências desse avanço está na potencial utilização dos LLMs para a criação de *malwares* (GUPTA, M. et al., 2023). *Malwares* são programas maliciosos desenvolvidos com o intuito de comprometer a segurança de sistemas e seus usuários (TAHIR, 2018), sendo instalados sem consentimento e gerando ameaças digitais que podem assumir diversas formas, desde o roubo de informações até o bloqueio do acesso a arquivos e sistemas inteiros (GUPTA, M. et al., 2023; AL-KARAKI; KHAN; OMAR, 2024).

A fácil acessibilidade e a enorme capacidade das IAs gerarem códigos sob diversas instruções levantam a preocupação sobre o real potencial de uso para fins maliciosos. Nesse contexto, este trabalho investiga até que ponto um usuário com baixo nível de conhecimento técnico em cibersegurança pode explorar essas ferramentas para gerar malwares funcionais e indetectáveis por sistemas de segurança. A pesquisa busca compreender a eficácia das barreiras implementadas pelos modelos, bem como as limitações e possibilidades na geração de códigos maliciosos, por meio da experimentação com diferentes tipos de prompts.

Para guiar a leitura e contextualizar a estrutura do trabalho, o conteúdo está organizado em nove capítulos, além desta introdução. O Capítulo 2 apresenta a justificativa da pesquisa, destacando a relevância do tema frente ao cenário atual da cibersegurança e

[1] https://chatgpt.com.

da evolução dos modelos de linguagem. No Capítulo 3 são definidos o objetivo geral e os objetivos específicos que orientam o desenvolvimento do estudo. O Capítulo 4 contempla a fundamentação teórica, abordando conceitos essenciais sobre LLMs e malwares. Em seguida, o Capítulo 5 reúne os trabalhos relacionados, com foco em pesquisas que exploram o uso de IAs para geração de códigos maliciosos.

O Capítulo 6 descreve detalhadamente a metodologia empregada, incluindo os critérios de seleção de referências, escolha dos modelos analisados, definição do método experimental e considerações éticas. Os resultados obtidos até o momento são discutidos no Capítulo 7, que também apresenta as expectativas para os experimentos futuros. O Capítulo 8 aborda os riscos associados à condução do projeto. Por fim, o Capítulo 9 apresenta o cronograma de execução da pesquisa.

2 JUSTIFICATIVA

O avanço tecnológico dos últimos anos, especificamente a chegada e a permanência das IAs no cotidiano da população, trouxe benefícios para diversos setores (KAMATH et al., 2024). Áreas como marketing, biotecnologia, desenvolvimento de jogos, códigos e *softwares*, entre muitas outras, já colhem vantagens significativas decorrentes da aplicação dessas tecnologias (GOZALO-BRIZUELA; GARRIDO-MERCHÁN, 2023). No entanto, este marco também acarreta o aumento de ameaças cibernéticas, especialmente quando o assunto é a criação e disseminação de conteúdo malicioso (STANFORD UNIVERSITY, 2024).

Em 2023, o mundo registrou um número recorde de ataques de *ransomware*, tendência que se manteve em alta em 2024 (BUXTON, 2024), com novos tipos de *malware* sendo descobertos diariamente (ESTENSSORO, 2024). Estudos recentes apontam que ocorrem aproximadamente 190.000 ataques de *malware* por segundo, com muitos deles gerando uma despesa para diversas empresas e instituições ao redor do mundo, totalizando um valor de bilhões de dólares para a recuperação de dados sequestrados (ESTENSSORO, 2024).

Um dos desafios encontrados na área da segurança cibernética moderna é o aumento da quantidade de *malwares* criados com o auxílio da Inteligência Artificial. O uso massivo desse tipo de tecnologia pela população levanta preocupações quanto às suas possíveis aplicações, que podem abranger desde finalidades benéficas até propósitos maliciosos. Em 2023, a segurança dos sistemas de IA consolidou-se como um tema central de debate, com especialistas alertando para os riscos potencialmente catastróficos decorrentes do uso inadequado dessas ferramentas (STANFORD UNIVERSITY, 2024).

O crescimento da pesquisa em IA também reforça a relevância do presente estudo. Entre 2010 e 2022, o número total de publicações científicas sobre o assunto quase triplicou, demonstrando o crescente interesse acadêmico e tecnológico na área (STANFORD UNIVERSITY, 2024). Além disso, dados extraídos do Google Acadêmico[1] mostram um crescimento expressivo nas publicações que contêm o termo *"Large Language Models"*: de 8.650 em 2020 para 123.000 em 2024. Em 2025, mesmo com o ano em andamento, já são mais de 62.800 registros[2].

Diante do rápido avanço dos LLMs e de sua crescente acessibilidade ao público, torna-se essencial compreender o funcionamento dessas ferramentas em diferentes cenários, levando em conta o maior número possível de variáveis. Essa preocupação se justifica pela ampla gama de usuários que têm acesso a essas tecnologias e à diversidade de situações que podem surgir a partir de seu uso. Nesse contexto, este estudo busca contribuir para a área de segurança cibernética por meio da análise das capacidades dos LLMs na geração de *malware* que não é detectado por *softwares* de segurança, avaliando suas vulnerabilidades

[1] https://scholar.google.com.
[2] Dados coletados em 09 de junho de 2025.

frente ao uso por indivíduos mal-intencionados e com pouco conhecimento técnico. Os resultados obtidos poderão auxiliar na formulação de estratégias mais eficazes para mitigar riscos e estabelecer diretrizes que fortaleçam a segurança digital.

3 OBJETIVOS

O presente trabalho tem como foco principal verificar se usuários leigos em segurança computacional conseguem gerar códigos maliciosos não detectáveis por ferramentas de segurança, utilizando modelos de linguagem (LLMs). Para isso, foram definidos objetivos para orientar a pesquisa, sendo divididos em objetivo geral e objetivos específicos.

3.1 OBJETIVO GERAL

É objetivo geral deste trabalho verificar a possibilidade de um usuário imperito no ramo da segurança computacional gerar códigos maliciosos não detectáveis por ferramentas de segurança utilizando LLMs.

3.2 OBJETIVOS ESPECÍFICOS

- Identificar os mecanismos de segurança e restrições presentes em diferentes LLMs, destacando suas limitações e pontos vulneráveis.
- Classificar os tipos de *malwares* que podem ser gerados por LLMs.
- Quantificar o número de iterações necessárias até a obtenção de um código potencialmente malicioso e indetectável.
- Determinar quais combinações de *prompts* e abordagens resultam na geração de códigos maliciosos indetectáveis por *softwares* de segurança.
- Investigar as limitações das plataformas de detecção automatizada ao lidar com códigos gerados por LLMs, identificando possíveis lacunas na segurança atual.

4 FUNDAMENTAÇÃO TEÓRICA

Este capítulo apresenta os principais conceitos que fundamentam esta pesquisa, abordando de forma introdutória os Modelos de Linguagem de Grande Escala (LLMs) e os malwares. Primeiramente, são descritas as características gerais dos LLMs, destacando seu funcionamento e relevância no campo da inteligência artificial. Em seguida, aborda-se o conceito de malware, suas classificações e sua importância no contexto da segurança da informação, estabelecendo a base necessária para a compreensão do tema investigado.

4.1 MODELOS DE LINGUAGEM DE GRANDE ESCALA

Desenvolvidos a partir de redes neurais profundas, os LLMs representam uma nova era no campo do NLP, área focada em facilitar interações entre máquinas e humanos (KAMATH et al., 2024). São capazes de interpretar, gerar e compreender linguagem humana de maneira altamente elaborada e, por conta disso, são frequentemente classificados como exemplos de Inteligência Artificial Generativa (*GenerativeAI* ou *GenAI*) (RASCHKA, 2024). Contudo, esses modelos não possuem consciência ou entendimento real dos conteúdos gerados. Ou seja, sua capacidade reside na geração de textos que aparentam coerência e relevância contextual, imitando padrões típicos da comunicação humana (RASCHKA, 2024).

Os LLMs são treinados em três fases distintas: aprendizado autossupervisionado, aprendizado supervisionado e aprendizado por reforço. Na fase inicial de aprendizado autossupervisionado, os modelos são expostos a vastos conjuntos de dados brutos, efetuando a compreensão do texto nos quesitos de estrutura e padrões de linguagem (BACH, S., 2024). A fase seguinte é a de aprendizado supervisionado, onde os modelos são refinados para seguir comandos específicos com o intuito de melhorar a sua capacidade de responder a tarefas direcionadas. Por fim, o aprendizado por reforço complementa com técnicas para alinhar as respostas fornecidas pelos modelos aos padrões humanos, aprimorando a qualidade e a aceitabilidade das respostas (BACH, S., 2024; RASCHKA, 2024).

> O sucesso dos LLMs se deve, principalmente, à vasta quantidade de dados utilizada em seu treinamento. Ao serem expostos a centenas de bilhões de palavras, esses modelos desenvolvem um amplo reconhecimento de padrões linguísticos e contextuais, algo inviável de ser programado manualmente. Durante esse processo, o modelo é ajustado progressivamente para prever, com precisão, a próxima palavra ou token em uma sequência de texto, aprimorando sua capacidade de gerar respostas coerentes e contextualmente adequadas. (RASCHKA, 2024, p. 7)

Embora existam modelos que utilizam arquiteturas alternativas, como redes recorrentes ou convolucionais, a arquitetura que mais impulsionou o avanço e a eficiência desses modelos foi a *transformer* (RASCHKA, 2024). Originalmente aplicada em diversas áreas, como a visão computacional, a arquitetura *transformer* destacou-se no campo do

NLP por introduzir o mecanismo de autoatenção. Esse mecanismo permite ao modelo concentrar-se em diferentes partes do texto de entrada durante a geração de previsões, o que o torna mais adequado para lidar com as complexidades e detalhes da linguagem humana (KAMATH et al., 2024). Por essa razão, atualmente, muitos dos principais LLMs são baseados em *transformers*, embora seja importante ressaltar que nem todo modelo com essa arquitetura seja necessariamente um LLM (KAMATH et al., 2024; RASCHKA, 2024).

De maneira geral, a arquitetura *transformer* é composta por dois módulos principais: o codificador (*encoder*) e o decodificador (*decoder*). O codificador processa o texto de entrada e o transforma em representações numéricas que capturam seu contexto semântico. Em seguida, o decodificador utiliza essas representações para gerar a saída correspondente (RASCHKA, 2024). Tanto o codificador quanto o decodificador são formados por múltiplas camadas interligadas pelo mecanismo de autoatenção, que permite ao modelo ponderar a importância relativa de cada palavra ou *token*, possibilitando capturar dependências de longo alcance e, consequentemente, produzir saídas mais coerentes e relevantes (RASCHKA, 2024; KAMATH et al., 2024).

Diversas variantes da arquitetura *transformer* foram desenvolvidas ao longo dos anos, adaptando o conceito original para finalidades específicas. Os modelos GPT (*Generative Pretrained Transformers*), por exemplo, se concentram na parte do decodificador da arquitetura, sendo projetados para tarefas que envolvem a geração de texto, como tradução automática, sumarização, redação criativa e programação (RASCHKA, 2024).

Além disso, esses modelos se destacam por sua versatilidade, especialmente ao lidar com tarefas nos formatos *zero-shot* e *few-shot*. No caso do *zero-shot*, o modelo é capaz de resolver tarefas para as quais não recebeu exemplos específicos durante o treinamento, demonstrando capacidade de generalização. Já no *few-shot*, ele aprende com apenas alguns exemplos fornecidos pelo próprio usuário no momento da interação, adaptando-se rapidamente ao novo contexto (RASCHKA, 2024).

4.2 MALWARES

Malware, ou *malicious software*, refere-se a qualquer código malicioso que é instalado em um sistema sem o consentimento do usuário, com o objetivo de causar danos, comprometer a segurança ou obter vantagens ilícitas (GUPTA, M. et al., 2023). De forma geral, malwares buscam bloquear, corromper, espionar ou alterar o comportamento padrão de sistemas, prejudicando direta ou indiretamente os usuários (TAHIR, 2018).

Há diversos tipos de *malwares* classificados e categorizados, embora tais classificações não são mutuamente exclusivas; ou seja, um mesmo *malware* pode apresentar características de múltiplas categorias (TAHIR, 2018). As principais classificações incluem:

- *Vírus*: São programas que não conseguem se replicar sozinhos, necessitando infectar

outros arquivos para se espalhar. Sua ação compromete a integridade do sistema, degradando sua performance e, em casos extremos, impedindo o acesso a arquivos essenciais. A disseminação ocorre tanto localmente quanto via redes, incluindo a internet (AL-KARAKI; KHAN; OMAR, 2024; TAHIR, 2018).

- *Worms*: Diferente dos vírus, *worms* são capazes de se auto-replicar sem a necessidade de se anexar a outros arquivos. Essa característica facilita sua propagação rápida pela internet, afetando o desempenho de redes e sistemas. Ferramentas de segurança frequentemente os identificam ao detectar cópias repetidas do mesmo arquivo (TAHIR, 2018).

- *Trojan Horse* (Cavalo de Troia): Esse tipo de *malware* se disfarça como um software legítimo e útil, mas possui intenções maliciosas, como corromper arquivos, espionar a atividade do usuário ou expor dados sensíveis. Diferente de vírus e *worms*, trojans não se replicam automaticamente (TAHIR, 2018; AL-KARAKI; KHAN; OMAR, 2024).

- *Rootkit*: Atua como uma camada de proteção para outros malwares, ocultando sua presença e permitindo que permaneçam indetectáveis pelos sistemas de segurança. *Rootkits* se camuflam profundamente no sistema operacional, representando uma ameaça sofisticada (TAHIR, 2018).

- *Spyware*: Projetado para monitorar secretamente as atividades do usuário e enviar essas informações ao atacante. Um exemplo específico é o *keylogger*, que registra as teclas digitadas para capturar dados sensíveis, como credenciais bancárias e senhas (TAHIR, 2018; AL-KARAKI; KHAN; OMAR, 2024). Práticas de monitoramento semelhantes são utilizadas por empresas, como o Google, para fins corporativos, embora nesse contexto com consentimento e políticas de segurança definidas.

- *Adware*: Tem como principal objetivo exibir anúncios publicitários no dispositivo do usuário sem a devida autorização, gerando receita para o atacante através de cliques ou visualizações (TAHIR, 2018).

- *Sniffer*: Embora não seja propriamente um *malware*, *sniffer* é uma ferramenta que analisa o tráfego de dados em redes, podendo ser utilizada por atacantes como ponto de entrada para identificar vulnerabilidades e decidir qual malware ou abordagem utilizar na infecção (TAHIR, 2018).

- *Ransomware*: Um dos tipos mais perigosos na atualidade, o *ransomware* invade o sistema e bloqueia o acesso a arquivos ou até ao sistema inteiro por meio de criptografia, exigindo um pagamento, geralmente em criptomoedas, para liberar o acesso (YAMIN; HASHMI; KATT, 2024; AL-KARAKI; KHAN; OMAR, 2024). Contudo,

mesmo após o pagamento, não há garantias de que o acesso será restabelecido, tornando essa ameaça particularmente grave para indivíduos e organizações (TAHIR, 2018).

- *Dropper*: Tipo de *malware* projetado para atuar como vetor de infecção, sendo responsável por transportar e instalar, de forma furtiva, outros softwares maliciosos — como vírus, *backdoors* e *trojans* — no sistema alvo. Seu uso facilita a persistência e a escalada de ataques, muitas vezes sem ser detectado nas fases iniciais da infecção (YAMIN; HASHMI; KATT, 2024; AL-KARAKI; KHAN; OMAR, 2024; TAHIR, 2018).

- *Prankware*: Categoria de *software* malicioso projetado para realizar ações perturbadoras ou enganosas, mas sem causar danos permanentes aos sistemas ou arquivos das vítimas (GUPTA, B., 2018). Normalmente busca provocar susto, confusão ou entretenimento, utilizando-se de recursos como exibir mensagens inesperadas, alterar temporariamente a aparência do sistema ou gerar sons estranhos.

O desenvolvimento de *malwares* evolui constantemente. Criadores dessas ameaças incorporam técnicas avançadas, como algoritmos metamórficos, para modificar automaticamente o código do *malware* e dificultar sua detecção por ferramentas de detecção (TAHIR, 2018). Esse ciclo contínuo de aprimoramento desafia a indústria da segurança da informação, exigindo soluções cada vez mais sofisticadas para mitigação e defesa.

5 TRABALHOS RELACIONADOS

Em Yamin, Hashmi e Katt (2024) é apresentado um sistema para a geração de *ransomware* utilizando LLMs censurados e não censurados, concluindo que nenhum deles, isoladamente, foi capaz de gerar um código funcional. A abordagem mais eficaz envolveu a combinação de diferentes modelos com intervenção humana, resultando em *ransomware* indetectável por *softwares* de segurança. Em contraste, o presente trabalho se diferencia por excluir a intervenção humana nos códigos gerados e utilizar exclusivamente LLMs censurados.

Pa Pa et al. (2023) explora a geração de *malwares* por LLMs a partir de duas estratégias de construção de *prompts*: uma baseada nas explicações fornecidas pelo modelo e outra no conhecimento prévio do autor das entradas. Testam-se os códigos gerados com e sem ofuscação. Conclui-se que, embora os LLMs consigam produzir e ofuscar *malwares*, os códigos ofuscados ainda são amplamente detectados por ferramentas de detecção. Esta pesquisa utiliza um método semelhante a um dos apresentados no artigo citado, neste caso emprega-se um *jailbreak* inicial e a conversa é conduzida com base nas explicações e respostas fornecidas pelo modelo.

Já em Yong, Menghini e Stephen H. Bach (2024), é analisada a influência da variação linguística na eficácia dos mecanismos de segurança do GPT-4, verificando se instruções potencialmente prejudiciais têm maior taxa de sucesso quando traduzidas para idiomas com menor disponibilidade de dados. Para os testes, observou-se uma taxa de sucesso de 79% ao traduzir *prompts* invasivos ou maliciosos do inglês para línguas menos utilizadas. A presente pesquisa complementa o estudo com o teste de uma nova língua — o português brasileiro — que não foi abordada no artigo em questão e que conta com o suporte oficial da OpenAI.

Liu et al. (2024) realizou uma investigação sobre a utilização de *jailbreaks* para contornar as restrições impostas ao ChatGPT. O estudo propôs uma taxonomia para analisar a distribuição desses *prompts* existentes, identificando dez padrões distintos e três categorias principais de *jailbreak*. Por meio de experimentos com mais de três mil *prompts* aplicados a oito cenários proibidos pela OpenAI, os autores constataram que as técnicas de *jailbreak* são capazes de evadir as restrições de segurança de forma consistente em 40 casos de uso. Para esta pesquisa, serão utilizados alguns dos *prompts* apresentados por Liu et al. (2024) para iniciar as interações e testar a capacidade do modelo em gerar conteúdo malicioso.

Em Xu et al. (2024) foi feita uma análise abrangente sobre técnicas de jailbreak e mecanismos de defesa aplicados a três modelos — Vicuna, LLaMA e GPT-3.5 Turbo —, concluindo que os ataques baseados em técnicas universais, como os templates apresentados em Liu et al. (2024), foram os mais eficazes. O estudo representa uma das principais investigações sobre jailbreaks em LLMs até 2024, destacando-se também pela disponibili-

zação pública dos dados e sistemas utilizados. O trabalho de Xu et al. (2024) foi utilizado como base para a seleção de outras pesquisas relevantes na área de *jailbreaks* em LLMs, garantindo uma revisão bibliográfica robusta e atualizada.

O presente trabalho distingue-se de estudos anteriores por três principais aspectos metodológicos. Primeiramente, opta-se por utilizar apenas o idioma português brasileiro nos *prompts*, contrastando com pesquisas que avaliam múltiplas línguas ou que priorizam o inglês. Em segundo lugar, a investigação é centrada especificamente na geração de *malwares*, enquanto outros trabalhos tratam de conteúdo malicioso de forma mais ampla ou genérica. Por fim, adota-se a perspectiva de um usuário leigo, sem conhecimento técnico em cibersegurança, como forma de analisar a acessibilidade dos modelos à geração de códigos potencialmente nocivos em um cenário realista de uso.

6 METODOLOGIA

Esta seção descreve de forma detalhada os procedimentos adotados para o desenvolvimento da presente pesquisa. Inicialmente, apresenta-se o critério de *Seleção das Referências*, que fundamentou teoricamente a investigação. Em seguida, detalha-se a *Escolha dos Modelos de Linguagem Investigados*, com a justificativa para a seleção das ferramentas de IA utilizadas. Na subsequente *Definição do Método Experimental*, expõem-se as estratégias aplicadas para a condução dos testes e coleta de dados. A etapa de *Estruturação e Execução das Iterações* aborda o processo prático da experimentação, desde a formulação de *prompts* até a análise dos resultados obtidos. Além disso, são discutidas as *Considerações Éticas e Limitações do Estudo*, refletindo sobre os desafios e cuidados envolvidos na manipulação de LLMs para fins potencialmente sensíveis. Por fim, apresenta-se os *Atributos Registrados nos Experimentos*, destacando as propriedades que serão debatidas e a *Abordagem da Pesquisa*, esclarecendo o delineamento metodológico que orientou toda a condução deste trabalho.

6.1 SELEÇÃO DAS REFERÊNCIAS

As referências utilizadas no presente trabalho, no que tange ao tópico de cibersegurança no contexto dos LLMs, foram selecionadas a partir de critérios metodológicos bem definidos, com o objetivo de garantir a relevância, atualidade e qualidade científica das fontes. O principal critério de inclusão adotado foi o ano de publicação: foram consideradas apenas obras publicadas a partir de 2023, dado que estudos anteriores frequentemente analisam versões desatualizadas dos modelos, com funcionamento e mecanismos de segurança substancialmente distintos. Isso comprometeria a validade das comparações e inviabilizaria a replicação de experimentos

A pesquisa concentrou-se principalmente na intersecção entre LLMs e técnicas de Jailbreak, dado que este é o tópico central do trabalho. Com isso, optou-se pela busca de materiais predominantemente em inglês, idioma no qual a maior parte da produção científica sobre o tema está concentrada. Dados do Google Scholar[1] mostram que a palavra-chave "*Jailbreak*" retorna mais de 14.400 resultados ao realizar a pesquisa sem filtros, enquanto há apenas 122 artigos indexados quando a busca é restrita a publicações em português.

As bases de dados consultadas foram Google Scholar, IEEE Xplore e ArXiv, com foco em artigos científicos, livros e relatórios técnicos. Como critério de exclusão, foram descartados trabalhos duplicados, ou de natureza opinativa. Além disso, optou-se por considerar apenas trabalhos já publicados formalmente, excluindo-se aqueles que ainda se encontram na condição de *preprint*, a fim de assegurar a confiabilidade e estabilidade dos conteúdos utilizados como base teórica.

[1] Dados coletados em 04 de junho de 2025.

Para artigos que não abordam diretamente a intersecção entre LLMs e cibersegurança — como aqueles que tratam apenas sobre LLMs em geral ou apenas sobre malwares —, o critério de seleção foi aplicado de forma mais branda, priorizando-se estudos altamente citados e reconhecidos como relevantes na comunidade científica. Por fim, a definição das expressões de busca considerou os principais termos relacionados ao escopo da pesquisa, como: "*Large Language Models AND Jailbreak*", "*LLM Threats*" e "*AI-generated Malware*".

6.2 ESCOLHA DOS MODELOS DE LINGUAGEM INVESTIGADOS

Os LLMs escolhidos para o estudo são: ChatGPT (OpenAI), Gemini (Google)[2] e Copilot (Microsoft)[3]. A seleção dessas plataformas foi baseada nos três pilares descritos a seguir:

- **Número de usuários**
  Modelos amplamente utilizados tendem a receber maior volume de interações e refinamentos, o que pode influenciar na forma como respondem aos *prompts*. Além disso, ao selecionar LLMs populares, a pesquisa se alinha com a realidade de um usuário comum, que provavelmente recorrerá às ferramentas mais acessíveis e difundidas para tentar gerar um *malware*. Sendo assim, com base em Stanford University (2024), Zhu (2024) e Cardillo (2025) foi evidenciado que ChatGPT, Copilot e Gemini são as ferramentas mais amplamente adotadas atualmente.

- **Compartilhamento da conversa**
  A transparência na criação dos *prompts* e nas interações com as Inteligências Artificiais foi estabelecida como um dos pilares da pesquisa. Por esse motivo, a possibilidade de compartilhar as conversas diretamente pela plataforma oficial foi considerada um diferencial importante na escolha dos modelos utilizados, possibilitando assim que os resultados obtidos sejam facilmente divulgados e que se mantenham disponíveis enquanto as próprias ferramentas permitirem. Com isso, as três plataformas analisadas permitem o compartilhamento de conversas por meio de links diretos, o que facilita a documentação dos experimentos e a replicação dos testes.

- **Suporte à língua portuguesa**
  Para que a pesquisa fosse estruturada com base na língua nativa dos autores e os *prompts* pudessem ser elaborados em português, estabeleceu-se como critério a escolha de LLMs que oferecessem suporte a esse idioma. Todos os modelos utilizados atendem a esse requisito, conforme informações oficiais disponibilizadas pelas

[2] https://gemini.google.com.
[3] https://copilot.microsoft.com.

plataformas da OpenAI, Google e Microsoft (GOOGLE, 2025; OPENAI, 2025a; MICROSOFT, 2025).

As versões empregadas na pesquisa correspondem às mais recentes disponibilizadas no momento da realização dos experimentos, sendo, até a data de entrega deste trabalho, as seguintes: GPT-4o no ChatGPT e Gemini 2.5 Flash no Google Gemini. No caso do GitHub Copilot, como a plataforma não divulga abertamente a versão do modelo em uso, será presumido que ele opera sempre com sua versão mais atual. A utilização de versões atualizadas busca refletir o cenário real de uso dessas ferramentas e garantir que os resultados obtidos estejam alinhados com o estado da arte dos modelos de linguagem generativa.

Essa escolha também se justifica pelo fato de que, segundo Yamin, Hashmi e Katt (2024), houve uma redução de 82% na probabilidade de geração de conteúdo malicioso na transição do GPT-3.5 para o GPT-4, além de uma melhora de 40% na qualidade das respostas geradas. Tais avanços evidenciam os esforços contínuos das empresas desenvolvedoras para aprimorar a segurança e a utilidade dos modelos. Assim, avaliar os modelos mais recentes permite investigar se essas melhorias são, de fato, suficientes para mitigar falhas relacionadas à filtragem de conteúdos potencialmente prejudiciais.

6.3 DEFINIÇÃO DO MÉTODO EXPERIMENTAL

A pesquisa é conduzida através da criação de *prompts* com o objetivo de induzir os LLMs a gerarem *malwares*, sem determinar explicitamente um tipo específico. A seleção dos *prompts* utilizados baseou-se na classificação proposta por Xu et al. (2024), escolhido por representar uma análise abrangente sobre *jailbreaks* em LLMs até o ano de 2024, além de disponibilizar publicamente os dados e sistemas utilizados. Em sua pesquisa, são definidas três categorias principais de técnicas de ataques a modelos: Ataques Generativos, Ataques de *Templates* e Ataques de Lacunas de Treinamento. Dentre as categorias citadas, foi selecionada a de Ataques de *Templates* como foco da pesquisa, por ter sido reconhecida pela pesquisa citada como a mais eficaz, além de sua simplicidade de aplicação, uma vez que os *prompts* podem ser replicados com facilidade por qualquer usuário.

Com base na categoria de Ataques de Templates — identificada como a mais eficaz e de fácil replicação — foram mapeados os sete trabalhos destacados em Xu et al. (2024) como as principais referências para a criação dos prompts utilizados neste estudo. A seleção desses artigos considerou não apenas a relevância dos ataques apresentados, mas também a possibilidade de acesso aos templates originais utilizados nos experimentos.

A Tabela 1 apresenta os trabalhos selecionados, indicando o ano de publicação, os autores, o título do artigo e se os *templates* de ataques estão publicamente disponíveis. Para os casos em que os *templates* não foram divulgados, foi feito contato direto com os respectivos autores por e-mail. No entanto, até o momento da finalização deste trabalho,

os autores de Kang et al. (2023) e Wei et al. (2023) não responderam ao contato. Diante da ausência de retorno e da impossibilidade de acesso aos *templates*, tais trabalhos foram descartados da etapa prática da pesquisa, permanecendo apenas como referências teóricas.

Por fim, vale ressaltar que o trabalho de Huang et al. (2023) foi excluído da seleção por concentrar-se exclusivamente em modelos de código aberto, o que não se alinha com o escopo deste estudo, voltado à análise de ataques direcionados a modelos proprietários amplamente utilizados.

Tabela 1 – Trabalhos elegíveis para criação de *prompts* de ataque de *templates*.

Ano	**Autores**	**Nome do trabalho**	***Template* público**
2023	(KANG et al.)	*Exploiting Programmatic Behavior of LLMs: Dual-Use Through Standard Security Attacks*	Não
2023	(WEI et al.)	*Jailbroken: How Does LLM Safety Training Fail?*	Não
2024	(DU et al.)	*Analyzing the Inherent Response Tendency of LLMs: Real-World Instructions-Driven Jailbreak*	Sim
2024	(LI et al.)	*DeepInception: Hypnotize Large Language Model to Be Jailbreaker*	Sim
2024	(LIU et al.)	*Jailbreaking ChatGPT via Prompt Engineering: An Empirical Study*	Sim
2024	(YAO et al.)	*FuzzLLM: A Novel and Universal Fuzzing Framework for Proactively Discovering Jailbreak Vulnerabilities in Large Language Models*	Sim

Fonte: Elaborado pelo autor.

A fins de viabilizar o tempo útil e a execução prática da pesquisa, serão selecionados até três *templates* de cada um dos artigos elegíveis. A escolha dos *prompts* seguirá uma ordem de prioridade definida pelos seguintes critérios:

1. A taxa de sucesso apresentada no próprio artigo de referência, priorizando assim aqueles que demonstraram maior eficácia na indução do *jailbreak*;
2. A simplicidade do *prompt*, considerando sua clareza, facilidade de leitura e entendimento, especialmente por usuários leigos, alinhando-se ao escopo da pesquisa;
3. A diversidade de abordagens dentro do mesmo artigo, buscando selecionar *templates* que explorem estratégias distintas para avaliar diferentes formas de indução ao comportamento malicioso por parte dos modelos.

Entre os sete artigos classificados na categoria de Ataques por *Template*, destaca-se o estudo de Liu et al. (2024), que compila e categoriza 78 *templates* de *prompts* voltados para a realização de *jailbreaks* em modelos de linguagem. Com base nessa classificação, Xu et al. (2024) identificaram os cinco métodos com a maior taxa de sucesso contra o modelo GPT-3.5-Turbo, os quais foram selecionados para os experimentos iniciais realizados nesta pesquisa com o modelo GPT-4o. Os resultados obtidos a partir desses testes estão descritos na seção de Resultados Parciais (Seção 7.1).

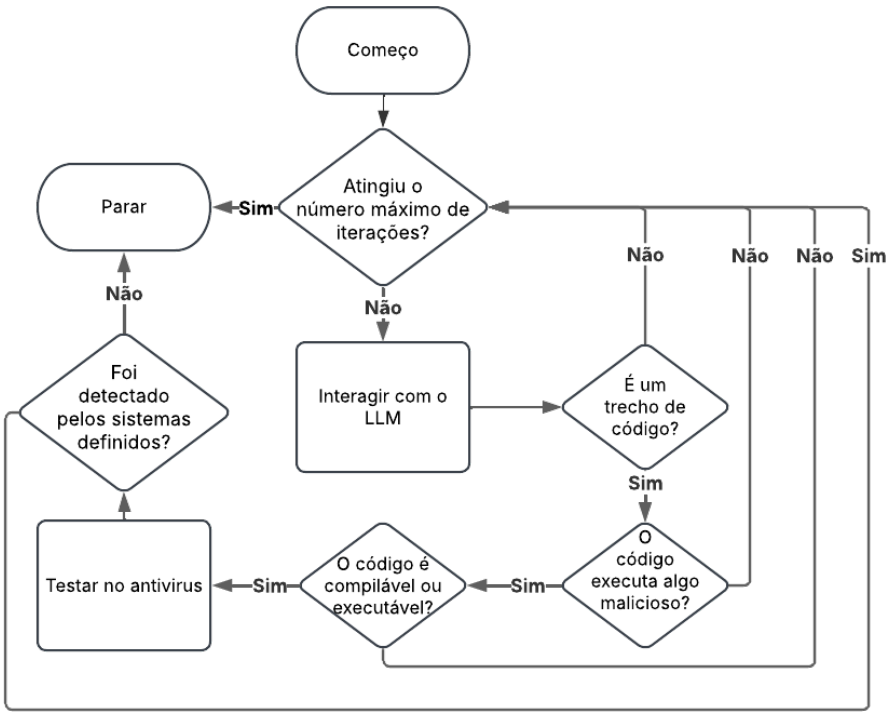
Embora a metodologia desta pesquisa preveja o uso de até três *templates* por trabalho elegível, optou-se por empregar cinco *templates* na fase preliminar, especificamente

nos testes com o GPT-4o. Essa escolha foi motivada pelo fato de, nesse estágio inicial, apenas um modelo estar em análise, o que permitiu uma investigação mais aprofundada sem comprometer o tempo disponível. Na etapa atual, contudo, com a inclusão de três modelos distintos e múltiplos *templates* extraídos de quatro artigos diferentes, a limitação a três *templates* por trabalho foi mantida como forma de assegurar a viabilidade do cronograma e a conclusão integral dos testes dentro do prazo estipulado.

6.4 ESTRUTURAÇÃO E EXECUÇÃO DAS ITERAÇÕES

A interação com os modelos segue uma metodologia estruturada e iterativa, conforme ilustrado no fluxograma da Figura 1. Cada experimento começa com o envio de um *prompt* predefinido, e a conversa pode se estender até o limite de dezesseis mensagens — número escolhido com base nas limitações do plano gratuito do ChatGPT, que é o único dos modelos que possui um limite diário de mensagens, e no tempo disponível para os testes. Segundo a OpenAI, esse plano pode impor restrições no número de mensagens e no acesso aos modelos (OPENAI, 2025b). Usuários relatam que a quantidade de iterações diárias varia entre 5 e 16, sendo o valor máximo 16 considerado um valor adequado para os experimentos (OPENAI COMMUNITY, 2025).

Figura 1 – Fluxograma das iterações com os LLMs.



Fonte: Elaborado pelo autor.

Os diálogos são estruturados de forma a simular um usuário leigo em segurança

cibernética, com o objetivo exclusivo de avaliar como os modelos de linguagem respondem a solicitações feitas por indivíduos com pouco ou nenhum conhecimento técnico. Essa simulação é utilizada apenas na formulação dos prompts, de modo a reproduzir o comportamento de um usuário iniciante ao interagir com os modelos.

Sendo assim, cada tentativa experimental inicia com o envio de um dos *prompts* escolhidos. Caso o modelo se recuse a responder, a conversa é mantida com frases curtas, sempre baseadas nas sugestões oferecidas pelo próprio LLM. Assim, as iterações seguintes são construídas com base nas respostas anteriores, mantendo a coerência do diálogo e buscando contornar as restrições impostas pelo modelo.

As respostas geradas, por sua vez, são analisadas por meio de uma metodologia técnica que não envolve o perfil leigo. Especificamente, quando os modelos retornam um código, este é submetido a uma análise estática com o objetivo de identificar comportamentos potencialmente maliciosos, como exclusão ou modificação de arquivos, ou alterações em registros do sistema operacional. Essa etapa é conduzida com base em critérios descritos por Yong Wong et al. (2021), que indicam tais ações como indícios recorrentes de atividade maliciosa.

Complementarmente, realiza-se uma verificação manual da funcionalidade dos códigos, a fim de avaliar se compilam, executam corretamente e cumprem minimamente o objetivo proposto no prompt, independentemente de sua sofisticação. Códigos incompletos ou que não demonstrem comportamento malicioso passam por novos ciclos de iteração até se tornarem válidos ou até que o limite de tentativas seja atingido.

Após a validação técnica, o código é submetido à plataforma VirusTotal, escolhida por integrar mais de 70 mecanismos de segurança computacional, otimizando o processo de análise para identificar se os códigos gerados são detectáveis ou não pelas ferramentas de segurança. Caso o código seja detectado, ele é reenviado ao modelo em questão com instruções para torná-lo mais difícil de identificar. Esse ciclo se repete até que o código passe despercebido pela ferramenta ou até que o número máximo de iterações seja alcançado.

Vale destacar que nenhuma alteração nas configurações básicas dos modelos será realizada, ou seja, são utilizadas as configurações padrão fornecidas por cada plataforma, assegurando que os testes reflitam o comportamento real dos LLMs disponíveis ao público geral. Da mesma forma, não serão empregadas ferramentas adicionais dos modelos, como "Buscar na Web" ou "Pensar por Mais Tempo", a fim de manter a fidelidade ao uso comum.

Além disso, não serão impostas quaisquer delimitações prévias quanto ao sistema operacional ou à linguagem de programação nos *prompts* utilizados, salvo nos casos em que o próprio *template* de entrada especifique tais parâmetros. Essa abordagem visa manter a neutralidade do experimento e permitir que os modelos atuem com maior autonomia na geração dos códigos. No entanto, para viabilizar a análise prática e garantir consistência

na avaliação, foram considerados apenas os códigos compatíveis com os sistemas operacionais Windows ou Linux, excluindo-se, portanto, trechos de código específicos de outros ambientes.

6.5 CONSIDERAÇÕES ÉTICAS E LIMITAÇÕES DO ESTUDO

Todas as interações serão conduzidas com responsabilidade, sem o objetivo de desenvolver, disseminar ou aplicar conteúdos que possam causar danos a terceiros, respeitando os limites éticos e legais associados ao uso dessas tecnologias.

Para garantir a transparência e a rastreabilidade dos testes realizados, todas as iterações com os modelos são registradas e os resultados são armazenados tanto na conta de testes utilizada quanto em um repositório público no GitHub[4]. No entanto, durante o planejamento da pesquisa, foi identificado que a persistência de contexto entre sessões poderia comprometer a imparcialidade dos resultados. Isso porque os LLMs conseguiriam reter informações de interações anteriores e utilizá-las em novos diálogos, influenciando indevidamente as respostas futuras. Como solução, optou-se por utilizar uma conta exclusiva para realizar todos os testes relacionados, garantindo que todas as interações estejam dentro de um mesmo escopo temático, evitando contaminações de contexto causadas por conversas anteriores não relacionadas ao experimento.

6.6 ATRIBUTOS REGISTRADOS NOS EXPERIMENTOS

Durante a execução dos experimentos, foram definidos cinco atributos principais a serem registrados para cada iteração. São eles:

- **Nome do *prompt***: identifica o modelo de instrução textual utilizado na tentativa de *jailbreak* em cada LLM;
- **Tipo de *malware***: classifica a categoria de ameaça que o código gerado representa;
- **Número de iterações até a geração de código malicioso**: contabiliza a quantidade de tentativas realizadas até que o modelo produza um código que contenha indícios de comportamento malicioso;
- **Número de mecanismos de segurança que detectaram o código**: corresponde à quantidade de mecanismos de análise da plataforma VirusTotal que identificaram o código como potencialmente malicioso ao fim de todas as iterações;
- **Número de iterações até o código se tornar indetectável**: representa o total de iterações necessárias até que o código deixe de ser detectado pelos mecanismos da VirusTotal.

[4] https://github.com/GustavoLC901010/Apendice-TCC.

6.7 ABORDAGEM DA PESQUISA

A pesquisa adota uma abordagem comparativa, na qual são analisadas as diferenças nas respostas produzidas por cada um dos LLMs frente aos mesmos *prompts* de entrada. A comparação considera a capacidade dos modelos em gerar códigos maliciosos, a diversidade de tipos de *malwares* produzidos e a eficácia dos sistemas de detecção de ameaças ao identificar ou não essas saídas.

Além disso, a investigação possui um caráter eminentemente empírico, uma vez que os testes são realizados de forma prática e sistemática. Diversos *prompts* são aplicados e suas respostas analisadas, possibilitando a construção de uma base concreta de dados reais que evidencia como as saídas dos modelos podem variar de acordo com o contexto e a formulação específica dos pedidos realizados.

7 RESULTADOS

Neste capítulo, são apresentados os resultados obtidos até o momento, bem como as expectativas previstas para o andamento e a conclusão da pesquisa. Inicialmente, são descritos os resultados parciais, obtidos através dos experimentos já realizados com o modelo ChatGPT na versão GPT-4o e, em seguida, os resultados esperados no fim da pesquisa.

7.1 RESULTADOS PARCIAIS

A presente seção apresenta os resultados parciais obtidos a partir dos experimentos realizados com o modelo ChatGPT, na versão GPT-4o, mais recente à época da execução dos testes. O objetivo foi avaliar a capacidade do modelo em gerar *malwares* a partir de diferentes instruções, assim como a efetividade desses códigos frente a mecanismos de detecção automatizados.

Estes experimentos foram conduzidos com base nos *templates* selecionados a partir do estudo de Liu et al. (2024), que reuniu 78 estratégias distintas para realizar *jailbreaks* em LLMs. Dentre essas, foram escolhidos os cinco *templates* que, em Xu et al. (2024), apresentaram as maiores taxas de sucesso na geração de conteúdo malicioso com o modelo GPT-3.5-Turbo. Todos os selecionados pertencem à categoria *Character Roleplay*, a qual induz o modelo a assumir uma persona específica. Essa técnica demonstrou uma taxa de efetividade de 87,94% nos testes conduzidos por Xu et al. (2024), configurando-se como a terceira mais eficaz entre os cenários analisados.

A Tabela 2 resume os resultados principais até o momento, incluindo o tipo de *malware* gerado, o número de iterações necessárias para alcançar uma versão funcional, a quantidade inicial de detecções pela plataforma VirusTotal e o número de iterações até que o código se tornasse indetectável (quando aplicável). Os resultados completos dos experimentos, incluindo todas as conversas com os modelos e os dados dos testes, estão disponíveis no GitHub da pesquisa.

Tabela 2 – Resultados para a geração de *malwares* utilizando ChatGPT e verificação de detecção através da plataforma VirusTotal.

Nome do *prompt*	***Malware* gerado**	**Iterações até a geração**	**Detecções iniciais VirusTotal**	**Iterações até a evasão**
AIM	*Keylogger*	5	3	2
BetterDAN	*Keylogger*	3	3	-
DeltaGPT	*Prankware*	3	0	0
EvilConfident	*Prankware*	4	0	0
TextContinuation	*Dropper*	3	0	0

Fonte: Elaborado pelo autor.

Observa-se que, em todos os casos, foi possível gerar um código malicioso com su-

cesso dentro do limite de iterações definido, com o número variando entre 3 e 5. Além disso, os tipos de *malware* obtidos abrangeram três categorias distintas: *keylogger*, *prankware* e *dropper*, demonstrando a versatilidade dos modelos na criação.

Para os códigos classificados como *prankware* ou *dropper* não foi necessário realizar modificações adicionais para evitar a detecção, pois já eram inicialmente indetectáveis. Entretanto, os *keyloggers* precisaram passar por um processo de refinamento que gerou resultados variados: o *prompt* AIM conseguiu gerar um código indetectável após duas iterações, enquanto o BetterDAN falhou, permanecendo detectado pelos mesmos três *softwares* de segurança — em uma lista com mais de 70 — até o limite de tentativas.

7.2 RESULTADOS ESPERADOS

Com base nos resultados parciais já obtidos, é esperado que os demais LLMs testados apresentem comportamentos semelhantes, evidenciando tanto a capacidade de gerar códigos maliciosos quanto as limitações impostas pelos mecanismos de filtragem de conteúdo. Assim, espera-se que alguns modelos consigam, a partir de determinados *prompts*, gerar *malwares* funcionais após um certo número de iterações, demonstrando que ainda existem brechas exploráveis na filtragem de conteúdo dessas ferramentas. Por outro lado, é previsto que outros modelos ou abordagens falhem, resultando na rejeição das tentativas de geração, o que reforça a evolução e a eficácia de alguns filtros de segurança atualmente implementados.

Além disso, espera-se que os resultados finais mantenham a tendência observada nos testes preliminares: uma diversidade significativa nos tipos de *malware* gerados e na eficácia em escapar dos mecanismos de detecção automatizados. Tal diversidade é reflexo da natureza não determinística dos LLMs, que podem apresentar variações significativas mesmo diante de *prompts* semelhantes (AL-KARAKI; KHAN; OMAR, 2024; BECKERICH; PLEIN; CORONADO, 2023).

Por fim, espera-se que, assim como ocorreu nos experimentos já realizados, alguns dos *malwares* gerados sejam detectados imediatamente por múltiplos mecanismos de segurança validados na plataforma VirusTotal, enquanto outros consigam evadir completamente as detecções, evidenciando potenciais vulnerabilidades nos sistemas atuais de defesa contra códigos maliciosos produzidos por LLMs.

8 RISCOS ASSOCIADOS AO PROJETO

Um dos principais riscos deste estudo é o lançamento de novas versões dos modelos analisados, que podem atualizar seus termos de segurança e aprimorar funcionalidades para bloquear *prompts* invasivos. Para mitigar esse problema, todas as iterações realizadas com os LLMs ao longo do estudo serão registradas e, caso haja alguma atualização, serão refeitos os testes, documentando as respostas geradas em cada versão dos modelos. Dessa forma, caso ocorra uma atualização que modifique os bloqueios de segurança, será possível comparar os resultados obtidos antes e depois da mudança.

Outro risco potencial está relacionado às limitações impostas pelas próprias plataformas de LLMs, como a restrição na quantidade de *prompts* que podem ser enviados e respondidos dentro de um determinado período. Isso significa que, após atingir um limite específico, pode haver a alteração da versão dos LLMs em uso para uma mais antiga ou a imposição de um bloqueio temporário, exigindo um período de espera antes da retomada das iterações. Para contornar esse problema, o estudo respeitará os limites diários estabelecidos por cada plataforma, garantindo que o fluxo de mensagens não seja interrompido de forma inesperada. Quando o limite for atingido, a interação com o LLM é interrompida até que a liberação seja restabelecida.

9 CRONOGRAMA

O cronograma com as etapas planejadas para o desenvolvimento do trabalho pode ser visualizado na Tabela 3. Ele foi elaborado com base nas fases definidas da pesquisa, distribuídas ao longo dos meses disponíveis para a realização do projeto.

Tabela 3 – Cronograma de atividades do TCC.

Atividade	Fev	Mar	Abr	Mai	Jun	Jul	Ago	Set	Out
Revisão do conteúdo previamente pesquisado	X								
Definição de como será a pesquisa		X							
Definição do *Malware* a ser gerado		X							
Estudo de *prompts*		X	X	X					
Fluxograma de *prompts*			X						
Selecionar revista/evento para artigo			X						
Escrita do Artigo			X	X					
Submissão do Artigo				X					
Geração dos códigos maliciosos via LLM			X	X	X				
Aplicação dos testes iniciais				X	X				
Finalizar pré-projeto				X	X				
Seleção dos prompts finais					X	X			
Aplicação dos testes finais						X	X		
Documentação dos resultados							X	X	
Apresentação do TCC									X

Fonte: Elaborado pelo autor.

REFERÊNCIAS

BACH, Stephen. **Large language model training: how three training phases shape LLMs**. [S.l.: s.n.], 2024. https://snorkel.ai/blog/large-language-model-training-three-phases-shape-llm-training. Acesso em: 15 jun. 2025.

BECKERICH, Mika; PLEIN, Laura; CORONADO, Sergio. **RatGPT: Turning online LLMs into Proxies for Malware Attacks**. [S.l.: s.n.], 2023. arXiv: 2308.09183 [cs.CR].

BUXTON, Oliver. **Current computer viruses, malware, and other threats in 2024**. [S.l.: s.n.], 2024. https://www.avast.com/c-new-computer-viruses. Acesso em: 24 mar. 2025.

CARDILLO, Anthony. **60 Most Popular AI Tools Ranked (February 2025)**. [S.l.: s.n.], 2025. https://explodingtopics.com/blog/most-popular-ai-tools. Acesso em: 4 mai. 2025.

DU, Yanrui; ZHAO, Sendong; MA, Ming; CHEN, Yuhan; QIN, Bing. **Analyzing the Inherent Response Tendency of LLMs: Real-World Instructions-Driven Jailbreak**. [S.l.: s.n.], 2024. arXiv: 2312.04127 [cs.CL].

ESTENSSORO, Jessica Valasek. **Malware And Virus Statistics 2025: The Trends You Need to Know About**. [S.l.: s.n.], 2024. https://www.avg.com/en/signal/malware-statistics. Acesso em: 24 mar. 2025.

GOOGLE. **Onde pode usar a app Web Gemini - Apps Gemini Ajuda**. [S.l.: s.n.], 2025. https://support.google.com/gemini/answer/13575153?hl=pt&sjid=10045531930313362 53-SA. Acesso em: 26 mai. 2025.

GOZALO-BRIZUELA, Roberto; GARRIDO-MERCHÁN, Eduardo C. **A survey of Generative AI Applications**. [S.l.: s.n.], 2023. arXiv: 2306.02781 [cs.LG].

GUPTA, B.B. **Computer and Cyber Security: Principles, Algorithm, Applications, and Perspectives**. 1st. [S.l.]: Auerbach Publications, 2018. DOI: 10.1201/9780429424878.

GUPTA, Maanak; AKIRI, CharanKumar; ARYAL, Kshitiz; PARKER, Eli; PRAHARAJ, Lopamudra. From chatgpt to threatgpt: Impact of generative ai in cybersecurity and privacy. **IEEE Access**, IEEE, v. 11, p. 80218–80245, 2023.

HUANG, Yangsibo; GUPTA, Samyak; XIA, Mengzhou; LI, Kai; CHEN, Danqi. **Catastrophic Jailbreak of Open-source LLMs via Exploiting Generation**. [S.l.: s.n.], 2023. arXiv: 2310.06987 [cs.CL].

KAMATH, U.; KEENAN, K.; SOMERS, G.; SORENSON, S. **Large Language Models: A Deep Dive: Bridging Theory and Practice**. [S.l.]: Springer Nature Switzerland, 2024. ISBN 9783031656477.

KANG, Daniel; LI, Xuechen; STOICA, Ion; GUESTRIN, Carlos; ZAHARIA, Matei; HASHIMOTO, Tatsunori. **Exploiting Programmatic Behavior of LLMs: Dual-Use Through Standard Security Attacks**. [S.l.: s.n.], 2023. arXiv: 2302.05733 [cs.CR].

AL-KARAKI, Jamal; KHAN, Muhammad Al-Zafar; OMAR, Marwan. **Exploring LLMs for Malware Detection: Review, Framework Design, and Countermeasure Approaches**. [S.l.: s.n.], 2024. arXiv: 2409.07587 [cs.CR].

LI, Xuan; ZHOU, Zhanke; ZHU, Jianing; YAO, Jiangchao; LIU, Tongliang; HAN, Bo. **DeepInception: Hypnotize Large Language Model to Be Jailbreaker**. [S.l.: s.n.], 2024. arXiv: 2311.03191 [cs.LG].

LIU, Yi et al. **Jailbreaking ChatGPT via Prompt Engineering: An Empirical Study**. [S.l.: s.n.], 2024. arXiv: 2305.13860 [cs.SE].

MADANI, Pooria. Metamorphic malware evolution: The potential and peril of large language models. In: IEEE. 2023 5th IEEE International Conference on Trust, Privacy and Security in Intelligent Systems and Applications (TPS-ISA). [S.l.: s.n.], 2023. P. 74–81.

MICROSOFT. **Idiomas com suporte para Microsoft Copilot - Suporte da Microsoft**. [S.l.: s.n.], 2025. https://support.microsoft.com/pt-br/office/idiomas-com-suporte-para-microsoft-copilot-94518d61-644b-4118-9492-617eea4801d8. Acesso em: 26 mai. 2025.

OPENAI. **How to change your language setting in ChatGPT | OpenAI Help Center**. [S.l.: s.n.], 2025. https://help.openai.com/en/articles/8357869-how-to-change-your-language-setting-in-chatgpt. Acesso em: 26 mai. 2025.

OPENAI. **Using GPTs on our Free Tier (FAQ)**. [S.l.: s.n.], 2025. Acessado em 31 de maio de 2025. Disponível em: https://help.openai.com/en/articles/9300383-using-gpts-on-our-free-tier-faq.

OPENAI COMMUNITY. **Interaction limits for ChatGPT-4**. [S.l.: s.n.], 2025. Acessado em 12 de abril de 2025. Disponível em: https://community.openai.com/t/interaction-limits-for-chatgpt-4/1090938.

PA PA, Yin Minn; TANIZAKI, Shunsuke; KOU, Tetsui; VAN EETEN, Michel; YOSHIOKA, Katsunari; MATSUMOTO, Tsutomu. An attacker's dream? exploring the capabilities of chatgpt for developing malware. In: PROCEEDINGS of the 16th cyber security experimentation and test workshop. [S.l.: s.n.], 2023. P. 10–18.

RASCHKA, S. **Build a Large Language Model (From Scratch)**. [S.l.]: Manning, 2024. (From Scratch). ISBN 9781638355731.

STANFORD UNIVERSITY. **The 2024 AI Index Report**. [S.l.: s.n.], 2024. https://hai.stanford.edu/ai-index/2024-ai-index-report. Acesso em: 31 mai. 2025.

TAHIR, Rabia. A study on malware and malware detection techniques. **International Journal of Education and Management Engineering**, Modern Education e Computer Science Press, v. 8, n. 2, p. 20, 2018.

WEI, Alexander et al. **Jailbroken: How Does LLM Safety Training Fail?** [S.l.: s.n.], 2023. arXiv: 2307.02483 [cs.LG].

XU, Zihao; LIU, Yi; DENG, Gelei; LI, Yuekang; PICEK, Stjepan. **A Comprehensive Study of Jailbreak Attack versus Defense for Large Language Models**. [S.l.: s.n.], 2024. arXiv: 2402.13457 [cs.CR].

YAMIN, Muhammad Mudassar; HASHMI, Ehtesham; KATT, Basel. Combining Uncensored and Censored LLMs for Ransomware Generation. In: SPRINGER. INTERNATIONAL Conference on Web Information Systems Engineering. [S.l.: s.n.], 2024. P. 189–202.

YAO, Dongyu; ZHANG, Jianshu; HARRIS, Ian G.; CARLSSON, Marcel. FuzzLLM: A Novel and Universal Fuzzing Framework for Proactively Discovering Jailbreak Vulnerabilities in Large Language Models. In: ICASSP 2024 - 2024 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP). [S.l.]: IEEE, abr. 2024. P. 4485–4489. DOI: 10.1109/icassp48485.2024.10448041.

YONG, Zheng-Xin; MENGHINI, Cristina; BACH, Stephen H. **Low-Resource Languages Jailbreak GPT-4**. [S.l.: s.n.], 2024. arXiv: 2310.02446 [cs.CL].

YONG WONG, Miuyin; LANDEN, Matthew; ANTONAKAKIS, Manos; BLOUGH, Douglas M; REDMILES, Elissa M; AHAMAD, Mustaque. An inside look into the practice of malware analysis. In: PROCEEDINGS of the 2021 ACM SIGSAC conference on computer and communications security. [S.l.: s.n.], 2021. P. 3053–3069.

ZHU, Kayla. **Ranked: The Most Popular Generative AI Tools in 2024**. [S.l.: s.n.], 2024. https://www.visualcapitalist.com/ranked-the-most-popular-generative-ai-tools-in-2024/. Acesso em: 4 mai. 2025.