Detecção de Ataques de Phishing por Meio de Modelos de Linguagem

Pedro M. M. Souza[1], **João V. S. Santos**[1], **Antonio M. B. Neto**[1], **Francisco V. J. Nobre**[1], **Alex F. R. Trajano**[2], **Rafael L. Gomes**[1]

[1]Universidade Estadual do Ceará (UECE)

{pedro.mikhael, jvs.santos, mozar.braga,

valderlan.nobre}@aluno.uece.br, rafa.lopes@uece.br

[2]Instituto Atlântico (IA)

alex_ferreira@atlantico.com.br

Abstract. *The increasing sophistication of phishing attacks, driven by generative AI, limits the effectiveness of traditional rule-based detection methods. This work presents VerificAI, a hybrid system for real-time phishing detection in email and SMS messages, integrating LLMs, SLMs, Retrieval-Augmented Generation, URL validation, and Active Learning. The system allows users to submit suspicious messages to a chatbot that provides an automated and explainable response. Experiments using the Enron Spam and SMS Spam Collection datasets compare cloud-based and local models. The results show that cloud-based LLMs achieve F1-scores of up to 97%, while local SLMs deliver competitive performance with lower latency and enhanced privacy.*

Resumo. *A sofisticação dos ataques de phishing, impulsionada por IA generativa, limita a eficácia de métodos tradicionais baseados em regras. Este trabalho apresenta o VerificAI, um sistema híbrido para detecção em tempo real de phishing em e-mails e SMS, integrando LLMs, SLMs, RAG, validação de URLs e Aprendizado Ativo. O sistema permite o envio de mensagens suspeitas a um chatbot com resposta automatizada e explicável. Experimentos com conjuntos de dados reais comparam modelos em nuvem e locais. Os resultados mostram que a versão VerificAI com LLMs em nuvem atinge até 97% de F1-score, enquanto a versão do VerificAI com SLMs locais oferece desempenho competitivo com menor latência e maior privacidade.*

1. Introdução

O cenário de segurança cibernética contemporâneo é marcado por uma sofisticação crescente das ameaças, impulsionadas em grande parte pelos avanços da inteligência artificial generativa, que permitem a criação de ataques cada vez mais convincentes e difíceis de detectar por métodos tradicionais [Costa et al. 2024, Brito et al. 2026]. Esse aumento na complexidade e no volume das ofensivas digitais tem sobrecarregado as defesas convencionais, tornando os usuários e as organizações mais vulneráveis a táticas de manipulação psicológica, aumentando os casos de vazamento de dados e sequestro de dados [Pimenta et al. 2025, Pimenta et al. 2024].

Dentre as diversas modalidades de crimes virtuais, o *phishing* destaca-se como uma das ameaças mais populares e eficazes, sendo uma forma de ataque de engenharia social, na qual agentes maliciosos enviam mensagens fraudulentas, tipicamente por e-mail ou *Short Message Service* (SMS), com o objetivo de induzir vítimas a revelar informações sensíveis, como credenciais de acesso, dados financeiros ou outros dados pessoais [Mendes et al. 2025]. Estas mensagens frequentemente se disfarçam de comunicações legítimas, explorando a confiança e a falta de atenção do usuário para alcançar seus objetivos [Souza et al. 2024].

Os avanços em modelos de linguagem natural têm revolucionado tarefas de compreensão e classificação de texto, incluindo a detecção de conteúdo malicioso. Estudos recentes exploram o potencial de *Large Language Model* (LLMs) e arquiteturas avançadas como DeBERTa para identificar padrões semânticos associados a *phishing* em *e-mails* e SMS, muitas vezes superando técnicas clássicas de aprendizado profundo em *benchmarks* públicos [Mahendru and Pandit 2024]. Tais modelos demonstraram capacidade de capturar nuances linguísticas e contextuais que escapam a abordagens baseadas apenas em atributos superficiais, embora ainda apresentem desafios relacionados à eficiência e robustez frente a ataques adversariais.

O uso de LLMs em contextos de *phishing* não se limita à classificação defensiva: também há evidências de que esses mesmos modelos podem ser explorados para gerar campanhas de *phishing* convincentes, o que aumenta ainda mais a urgência de mecanismos robustos de detecção [Schmitt and Flechais 2024]. Por outro lado, *frameworks* dedicados como o *ChatSpamDetector* demonstraram que LLMs podem fornecer classificações altamente precisas e explicáveis de *e-mails* suspeitos, facilitando a tomada de decisão por parte de usuários ou administradores de sistemas de segurança [Koide et al. 2024]. Além disso, pesquisas voltadas à engenharia de *prompts* explorando capacidades intrínsecas de LLMs para tarefas de classificação, sem a necessidade de fine-tuning pesado, mostram que tais modelos podem ser adaptados à detecção de phishing com desempenho competitivo e baixo custo de treinamento [Hasan et al. 2025].

Apesar desses progressos, lacunas importantes permanecem: a maioria dos trabalhos ainda se concentra em avaliações isoladas de modelos ou em cenários restritos de experimentação offline, sem abordar a integração prática de modelos de linguagem em fluxos de trabalho interativos e operacionais. Em especial, são escassas as soluções que permitam ao usuário final encaminhar mensagens suspeitas e receber respostas automatizadas e acionáveis em tempo quase real. Adicionalmente, poucos estudos investigam o uso de modelos menores, como *Small Language Models* (SLMs) [Wang et al. 2024], que podem oferecer melhor custo-benefício, menor latência e maior viabilidade de implantação em ambientes produtivos.

Dessa forma, este trabalho propõe um sistema baseado em modelos de linguagem para detecção de *phishing* em mensagens de e-mail e SMS, chamado VerificAI. No VerificAI os usuários podem encaminhar mensagens suspeitas para um serviço (via API, email ou chatbot) que analisa o conteúdo e retorna um veredito automatizado e acionável. A proposta investiga tanto o uso de LLMs quanto de SLMs, analisando o equilíbrio entre capacidade semântica, custo computacional, latência e viabilidade operacional.

O fluxo de funcionamento do VerificAI inicia-se com a captura e o pré-

processamento de e-mails ou SMS encaminhados pelo usuário para um chatbot, onde o texto é higienizado e as URLs são extraídas. O Núcleo de Inteligência coordena o processo realizando a validação determinística de URLs e a recuperação de contexto histórico através de *Retrieval-Augmented Generation* (RAG) [Xu et al. 2024], que busca no banco os exemplos de ataques mais similares à mensagem atual. Essas informações consolidam um prompt dinâmico enviado a um modelo de linguagem (LLM ou SLM), que interpreta os sinais de engenharia social e gera um veredicto de risco (SAFE, SUSPICIOUS ou MALICIOUS) acompanhado de uma explicação técnica. Por fim, o sistema utiliza Aprendizado Ativo, permitindo que o feedback direto do usuário sobre a precisão da análise seja imediatamente incorporado à base de conhecimento, refinando as detecções futuras.

Os experimentos do trabalho avaliaram o sistema VerificAI utilizando subconjuntos de 2.000 amostras dos conjuntos de dados públicos Enron Spam e SMS Spam Collection. Foram testados seis modelos de linguagem em diferentes ambientes: modelos em nuvem (como a família Gemini e DeepSeek) e modelos locais via Ollama (Llama 3 e Gemma), analisando métricas de classificação e tempo de inferência. Os resultados demonstraram que o modelo Gemini 3 Flash Preview obteve o melhor desempenho, com F1-Scores de até 97,14% para e-mails e 88,89% para SMS. Em contrapartida, os modelos menores (SLMs) executados localmente, embora menos precisos, apresentaram latência significativamente menor (chegando a apenas $1,28s$ por mensagem) e maior garantia de privacidade. Além dos *benchmarks*, um estudo de caso prático na UECE confirmou a eficácia da solução em um cenário real de engenharia social, atingindo 94% de assertividade.

O restante deste artigo está organizado da seguinte forma: a Seção 2 apresenta os trabalhos relacionados; a Seção 3 descreve a arquitetura e o funcionamento do sistema proposto; a Seção 4 detalha a metodologia experimental; a Seção 5 discute os resultados obtidos; e, por fim, a Seção 6 apresenta as conclusões e direções para trabalhos futuros.

2. Trabalhos Relacionados

Esta seção apresenta uma análise crítica de estudos acadêmicos que investigam a detecção de phishing, cujas abordagens evoluíram de métodos baseados em regras e aprendizado de máquina tradicional para técnicas de deep learning e, mais recentemente, LLMs. A revisão das abordagens existentes destaca as principais contribuições, limitações e lacunas de pesquisa, que fundamentam a motivação e a justificativa da proposta apresentada neste artigo.

Mendes et al. [Mendes et al. 2025] apresentam o *MeAJOR Corpus*, um *dataset* multi-origem para detecção de *phishing em e-mails*, construído a partir da fusão de diversos repositórios públicos. O trabalho enfatiza a importância da qualidade e diversidade dos dados para o desempenho de modelos de aprendizado de máquina, avaliando classificadores sobre diferentes conjuntos de *features* textuais e estruturais. Entretanto, a abordagem depende fortemente de engenharia manual de atributos e não explora modelos baseados em LLMs nem cenários de interação direta com o usuário, como sistemas conversacionais para verificação de mensagens suspeitas.

Afonso et al. [Afonso et al. 2025] investigam de forma aprofundada como problemas de qualidade em *datasets* de *phishing* impactam a generalização de modelos de detecção, especialmente em tarefas baseadas em URLs. O estudo demonstra que

duplicações, conflitos de rótulos e sobre-representação de domínios levam a avaliações excessivamente otimistas quando se utiliza apenas validação dentro do *dataset*. Embora o trabalho traga contribuições importantes para a compreensão das limitações dos *benchmarks* atuais, seu foco está restrito à detecção baseada em URLs, não abordando diretamente o conteúdo semântico de mensagens de e-mail ou SMS, nem o uso de LLMs em cenários interativos com usuários finais.

Wang et al. [Wang et al. 2025] apresentam o SmishX, um sistema explicável para detecção de *phishing* em mensagens SMS baseado em agentes LLM. O modelo combina o texto curto das mensagens com contexto externo, como informações de URLs e domínios, para melhorar a capacidade de raciocínio do LLM, além de fornecer explicações compreensíveis ao usuário final. Estudos com usuários indicam que as explicações aumentam a confiança e a capacidade de tomada de decisão. Apesar desses avanços, a solução depende de fontes externas de contexto, o que pode introduzir latência adicional, e não contempla a integração simultânea de e-mails e SMS em um único fluxo de atendimento automatizado.

Li et al. [Li et al. 2025] propõem o *FedPhishLLM*, um *framework* que combina LLMs com aprendizado federado para detecção de phishing. O sistema permite o treinamento colaborativo entre múltiplos participantes sem compartilhamento de dados brutos, apresentando bons resultados de acurácia e robustez frente a ataques evasivos. Contudo, o trabalho foca principalmente no processo de treinamento e na arquitetura de aprendizado federado, não explorando a experiência do usuário final nem a implementação de um chatbot operacional capaz de receber mensagens encaminhadas e retornar um veredito por e-mail ou SMS em tempo quase real.

Embora os trabalhos existentes avancem significativamente na detecção de *phishing* com aprendizado de máquina, *deep learning* e LLMs, ainda há uma lacuna na literatura quanto a soluções que transponham esses modelos para cenários operacionais e interativos. Especificamente, poucos estudos investigam o uso de SLMs como alternativa viável para privacidade e baixa latência ou a aplicação de RAG para fornecer contexto dinâmico e mitigar alucinações. A proposta deste artigo busca preencher essas lacunas ao apresentar um sistema unificado para e-mail e SMS que combina a capacidade semântica de modelos generativos com uma arquitetura de recuperação contextual e aprendizado ativo.

3. Proposta

Este trabalho apresenta o *VerificAI*, um sistema de segurança cibernética desenvolvido para proteger fluxos de e-*mails* contra táticas de *phishing* e engenharia social. Diante das limitações dos detectores estáticos atuais, a solução estabelece uma arquitetura híbrida que combina a capacidade interpretativa de LLMs e SLMs com o rigor dos métodos determinísticos. O elemento principal desta arquitetura é a aplicação de RAG, estruturada sobre o *framework LangChain* (v1.0.0)[1] e o banco vetorial ChromaDB (v1.3.5)[2]. Ao converter o histórico de ameaças em uma base de conhecimento dinâmica, o sistema permite a identificação de padrões em ataques inéditos, transcendendo as limitações das verificações binárias convencionais.

[1] `https://www.langchain.com/`
[2] `https://docs.trychroma.com/`

O *VerificAI* adota um fluxo de *Active Learning*, no qual a base de conhecimento é continuamente aprimorada a partir do *feedback* direto do usuário. Além da automação técnica, como a validação de URLs via *Google Safe Browsing*, o sistema explora a capacidade de raciocínio de LLMs, como o Gemini, para realizar uma análise minuciosa do conteúdo. Ao priorizar a análise comportamental, o sistema identifica táticas de engenharia social que conseguem contornar ferramentas tradicionais, as quais dependem exclusivamente de listagem prévia em *blocklists*.

A arquitetura contribui para o estado da arte ao tratar a comunicação de forma contextual e abrangente, em vez de processar mensagens como eventos isolados. A integração da memória via RAG ainda possibilita que o sistema gere respostas explicáveis ao destinatário. Com isso, a proteção deixa de ser uma barreira meramente reativa para se tornar uma ferramenta de segurança assistida, auxiliando o usuário a compreender as táticas de fraude às quais está exposto.

3.1. Arquitetura do Sistema

O *VerificAI* foi desenvolvido como uma solução modular e escalável, como ilustrado na Figura 1, operando independentemente da plataforma de origem da mensagem e através de uma arquitetura de microsserviços. A proposta adota um fluxo híbrido que combina validação determinística, recuperação de informação contextual e capacidade generativa.

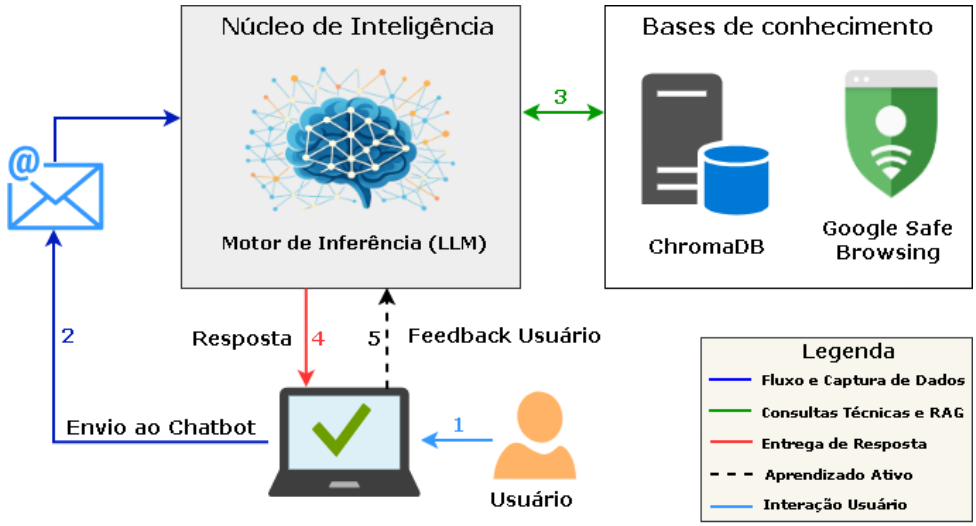


Figura 1. Visão Geral do Sistema.

O fluxo de execução foi organizado em dois grandes ambientes funcionais: Núcleo de Inteligência e Bases de Conhecimento. A execução segue uma sequência na qual a mensagem é coletada, processada pelo núcleo central, que integra as informações de reputação das URLs presentes no texto (*Google Safe Browsing*) com a recuperação do contexto histórico, sendo, por fim, submetida à inferência do modelo generativo. Embora independentes, cada módulo foi projetado para convergir sob uma coordenação central, operando de forma articulada para formar um sistema coeso.

O fluxo operacional do *VerificAI* tem início com a interação direta do usuário na interface do sistema (1), na qual a mensagem é submetida à análise. Em seguida, essa informação é capturada e encaminhada ao Núcleo de Inteligência (2), caracterizando a

etapa de recepção e aquisição de dados. Atuando como porta de entrada do sistema, esse componente centraliza o recebimento dos *e-mails* submetidos ao *chatbot*.

Essa fase corresponde à camada de aquisição e pré-processamento, sendo responsável pela higienização do conteúdo textual. Nela, *tags* HTML irrelevantes e caracteres de formatação são removidos, preservando-se apenas o corpo da mensagem e seus metadados para análise posterior. Adicionalmente, o processo de captura é executado em intervalos programados de um minuto, funcionando como um mecanismo de *rate limiting*, com o objetivo de evitar a exaustão de quotas da API e garantir a estabilidade operacional do sistema antes do encaminhamento aos módulos centrais de processamento. O processamento inteligente e a tomada de decisão concentram-se nos seguintes módulos:

- **Núcleo de Inteligência:** Atua como o centro de coordenação do sistema. Desenvolvido em Python com suporte das bibliotecas *LangChain* e *Google Generative AI*, este módulo é responsável pela engenharia de contexto. Em vez de realizar a classificação de risco diretamente, sua função central consiste na construção dinâmica do *prompt*. O núcleo consolida a mensagem original, integra os resultados determinísticos da validação técnica de URLs via *Google Safe Browsing* (3) e os vetores de similaridade histórica recuperados pelo mecanismo de RAG. O contexto resultante é submetido a um modelo generativo de linguagem que interpreta a correlação entre os indicadores e produz uma saída estruturada em formato JSON estrito, assegurando a integração com os demais módulos do sistema.
- **Bases de Conhecimento:** Este ambiente fornece o suporte necessário para mitigar alucinações e aumentar a precisão das inferências. Ele integra dois serviços distintos:
    1. **Google Safe Browsing:** Utilizado para a validação técnica de URLs, verificando a reputação de domínios e identificando endereços previamente catalogados como maliciosos.
    2. **ChromaDB (Memória Vetorial):** Responsável pela implementação do método de RAG. O modelo ***BAAI/bge-small-en-v1.5*** [Xiao et al. 2024] é empregado para transformar as mensagens em vetores de 384 dimensões. A escolha desse modelo fundamenta-se em sua elevada eficiência computacional e desempenho semântico, uma vez que ocupa posições de destaque no *Massive Text Embedding Benchmark* (MTEB) para modelos de porte reduzido. Sua arquitetura compacta viabiliza inferência de baixa latência na infraestrutura local proposta, mantendo a densidade semântica necessária para a recuperação de contextos relevantes armazenados no ChromaDB ($k = 3$), os quais subsidiam a tomada de decisão do modelo generativo.

Finalmente, o ciclo operacional é concluído com a devolução da resposta à interface do usuário (4), onde o resultado da inferência é apresentado de forma estruturada. A arquitetura incorpora um mecanismo de Aprendizado Ativo, no qual o usuário pode validar ou corrigir a inferência produzida. Esse *feedback* (5) é utilizado para atualizar dinamicamente a base de conhecimento: o Núcleo de Inteligência transforma o par (mensagem original + veredito corrigido) em um vetor de *embedding*, armazenado de forma persistente no ChromaDB. Em inferências subsequentes, esse novo exemplo pode ser recuperado pelo mecanismo de RAG, atuando como contexto imediato para aprimorar a capacidade de generalização do modelo frente a novas ameaças semanticamente semelhantes.

3.2. Detecção de Ameaças com IA Generativa e RAG

A efetividade do *VerificAI* está em sua capacidade de superar as restrições dos detectores fundamentados em assinaturas estáticas. Ao invés de procurar somente palavras-chave proibidas, o sistema emprega a habilidade de interpretação semântica das LLMs combinada com uma memória contextual. O processo de inferência consiste em três etapas sequenciais: vetorização da mensagem, recuperação contextual e análise generativa.

1. Vetorização e Recuperação: Cada nova mensagem recebida é convertida em sua representação vetorial e imediatamente comparada com a base de dados do ChromaDB. A recuperação dos exemplos mais similares atua como uma abordagem de *Few-Shot Learning*, aprimorando o modelo com exemplos práticos de ameaças e falsos positivos. Isso reduz a necessidade de uma ampla janela de contexto e viabiliza a identificação de ataques que ainda não foram identificados.

2. Engenharia de *Prompt* e *Digital Guardian*: A classificação final ocorre via requisições REST aos modelos generativos. Para assegurar a integração sistêmica, utiliza-se *Persona Prompting* instruindo a IA a atuar como especialista em cybersegurança e exigindo a resposta em formato JSON estrito. O *prompt* é construído dinamicamente concatenando: (i) a mensagem original; (ii) o status do *Safe Browsing*; e (iii) os exemplos do RAG, cujos textos são truncados em 300 caracteres para otimização de *tokens*. A Figura 2 apresenta essa estrutura lógica.

<ROLE>
You are the "Digital Guardian,"a cybersecurity AI by LARCES (UECE).

<MISSION>
1. **PROTECT:** Analyze to determine risk (SAFE, SUSPICIOUS, MALICIOUS).
2. **INTERACT:** If SAFE, be a helpful assistant.

<CONTEXT>
- Msg: {INPUT_TEXT}
- SafeBrowsing: {URL_STATUS}
- RAG: {RETRIEVED_VECTORS}

<INSTRUCTIONS>
1. Methodical Analysis (Vector of Attack):
- **URL Analysis:** If "PERIGOSO"→ **MALICIOUS**.
- **CRITICAL CHECK (Override Rule):** The "SAFE"result from SafeBrowsing **CAN BE OVERRIDDEN** by contextual analysis.
- **Domain Impersonation:** If message impersonates official entity (e.g. UECE, Banks) but link uses unrelated domain → **MALICIOUS** (regardless of technical status).
- **Social Eng.:** Identify triggers of Greed, Urgency, Authority.
- **History (RAG):** If similar vectors are PHISHING → increase risk.

2. Output Format (Strict JSON):
Return: `risk_level`, `analysis_details`, `user_response` (in Portuguese).

Figura 2. Estrutura do *prompt* do sistema usado no VerificAI, definindo explicitamente as regras de sobreposição para classificação de risco.

3. Análise Metódica e Cadeia de Pensamento (*Chain-of-Thought*): O *prompt* estabelece uma sequência analítica rigorosa para reduzir a ocorrência de falsos negativos. A instrução exige que a LLM/SLM execute uma Verificação de Impersonificação de Domínio, um passo fundamental em que o modelo confirma se o *link*, embora seja tecnicamente considerado seguro pelo *Safe Browsing*, corresponde à entidade mencionada na mensagem. Conforme detalhado nas diretrizes da Figura 2, caso seja identificada uma discrepância ou se houver gatilhos de engenharia social confirmados pelo histórico,

o modelo é instruído de forma explícita a sobrescrever o veredito técnico, aumentando a classificação de risco com base exclusivamente na evidência contextual.

3.3. Classificação de Risco

A fim de assegurar a compatibilidade com sistemas de defesa automatizados e proporcionar clareza ao usuário final, a saída do modelo segue um formato JSON rigoroso, conforme detalhado nas instruções de saída do sistema. A ameaça é classificada em uma categoria de risco, cujos critérios de decisão são estabelecidos da seguinte maneira:

- **SAFE (Seguro):** Concedido quando a avaliação técnica da URL não indica ameaças e o conteúdo semântico não revela sinais de engenharia social. O sistema produz uma resposta amigável, certificando a autenticidade da mensagem.
- **MALICIOUS (Malicioso):** Classificação atribuída em duas situações: (1) Detecção definitiva de URL em *blocklists* (por meio do *Safe Browsing*), ou (2) Reconhecimento de alta confiança de estratégias de *phishing*, sem levar em consideração o *status* da URL. O sistema alerta o usuário do perigo real da mensagem, que pode coletar dados sensíveis.
- **SUSPICIOUS (Suspeito):** Esta categoria trata da área crítica da detecção. É concedida quando existem indícios contraditórios, como uma URL aparentemente limpa, mas acompanhada de um texto que possui um forte apelo emocional ou de urgência. Nessas situações, o sistema adota uma abordagem cautelosa, informando o usuário sobre o possível risco.

3.4. Aprendizado Ativo e Adaptação via RAG

Uma característica fundamental dessa arquitetura é a incorporação de um ciclo de *feedback* especializado, referido neste estudo como Aprendizado Ativo. Esse mecanismo possibilita que o sistema reduza a obsolescência do conhecimento do modelo pré-treinado, ajustando-se a novas campanhas de *phishing* ou a padrões de comunicação legítimos específicos da organização em tempo real, sem precisar realizar o custoso re-treinamento.

A operação é realizada na interface de pós-análise. O sistema funciona como um instrumento de assistência: a IA executa uma análise técnica estruturada e oferece uma resposta ao usuário, que, com essa nova compreensão, pode confirmar a inferência com base em seu próprio contexto (por exemplo, indicando falsos positivos de remetentes institucionais reconhecidos). Para reduzir o risco de envenenamento da base de conhecimento (*data poisoning*) por usuários mal-intencionados, esse *feedback* é vetorizado e armazenado no ChromaDB de forma individualizada. Isso garante que a interação temporária refine as inferências por meio do RAG de forma segura.

Os dados são armazenados de forma otimizada para recuperação semântica, onde o conteúdo da mensagem é indexado como o vetor principal e a classificação correta é associada como metadado. O formato de armazenamento segue a estrutura direta implementada no código do sistema:

Exemplo de Vetor Armazenado (ChromaDB):

```
[PAGE_CONTENT]: "Temos o prazer de informar que a
lista de estudantes selecionados para a bolsa de
estudos já está disponível online..."
[METADATA]: {"label":  "MALICIOUS"}
```

Quando uma nova mensagem é recebida, o mecanismo de RAG busca neste banco vetorial as mensagens mais similares semanticamente. Ao encontrar exemplos históricos parecidos que possuem o metadado `"label": "MALICIOUS"`, o modelo de linguagem utiliza essa informação como um guia de referência (*Few-Shot Prompting*) para classificar a nova ameaça com a mesma severidade.

Quando uma nova mensagem similar é processada, o mecanismo de RAG recupera este bloco. Ao ler que um humano validou como **"CORRETO (Sim)"** uma análise que apontava risco *MALICIOUS* para aquele padrão de texto e remetente, a LLM/SLM reforça sua confiança para classificar a nova ameaça da mesma forma (*In-Context Learning*). O inverso também ocorre: se o feedback fosse "Não"(Incorreto), o modelo aprenderia a tratar aquele padrão como um falso positivo no futuro.

4. Experimentos

Esta seção descreve a configuração experimental adotada para avaliar a eficácia do *VerificAI* em cenários realistas de detecção de ameaças baseadas em linguagem natural. O estudo foi conduzido com o objetivo de responder a duas questões centrais: (i) se uma arquitetura baseada em RAG apresenta desempenho superior à detecção fundamentada exclusivamente no conhecimento pré-treinado dos modelos de linguagem; e (ii) se SLMs executados localmente podem substituir LLMs em nuvem sem impacto significativo na capacidade de detecção. Para isso, realizou-se um estudo comparativo envolvendo diferentes modelos, bases de dados e ambientes de execução, assegurando condições controladas e reprodutíveis. Ressalta-se que o código-fonte desenvolvido, bem como os conjuntos de dados utilizados nos experimentos, estão disponíveis no repositório do projeto[3] , juntamente com as instruções necessárias para a plena reprodutibilidade dos resultados.

Os experimentos utilizaram dois conjuntos de dados amplamente consolidados na literatura: o *SMS Spam Collection* [4] e o *Enron Spam Dataset* [5]. A opção por estes *benchmarks* públicos em vez das bases utilizadas nos trabalhos relacionados como o de Koide et al [Koide et al. 2024], justifica-se pela necessidade de transparência e reprodutibilidade, uma vez que o uso de dados particulares nesses estudos inviabiliza a validação externa. Além disso, o *dataset* Enron contém mensagens comuns de e-mails, enquanto o *dataset* SMS Collection inclui mensagens típicas de SMS. Essa dualidade é essencial para validar a robustez da arquitetura proposta frente a diferentes canais de ataque.

Considerando os custos computacionais e a latência associados à inferência com modelos de grande porte, foi selecionado um subconjunto aleatório e representativo de cada base, totalizando 2.000 amostras. Desse montante, foram utilizados 1.200 registros do *dataset* de e-mails, representando 6,43% de sua base total, e 800 registros do *dataset* de SMS, o equivalente a 14,36% do total, preservando-se a proporção original entre instâncias legítimas e maliciosas. Todas as mensagens passaram por um processo de higienização textual padronizado, descrito na Seção 3, resultando em um conjunto balanceado e adequado para análises comparativas estatisticamente significativas.

A avaliação contemplou seis modelos de linguagem, abrangendo diferentes tamanhos de parâmetros, arquiteturas e estratégias de implantação. Como *baseline*, foi

[3]https://github.com/LarcesUece/VerificAI

[4]https://archive.ics.uci.edu/dataset/228/sms+spam+collection

[5]https://www.kaggle.com/datasets/marcelwiechmann/enron-spam-data

utilizado o modelo Gemini 2.5 Flash por meio de sua API oficial (Google AI Studio), escolhido por ser o modelo mais custo-benefício no momento da pesquisa. Ale disso, foram usados os modelos DeepSeek-v3.2 e Gemini 3 Flash Preview via API oficial do Ollama Cloud, incluído para explorar o limite superior de desempenho oferecido por arquiteturas mais recentes. Adicionalmente, três modelos foram executados localmente via Ollama (Llama 3 (8B), Llama 3.2 (3B) e Gemma 3 (12B), com o objetivo de analisar a relação entre desempenho, consumo de recursos e viabilidade de implantação em infraestrutura própria.

Para garantir a consistência experimental, a orquestração do sistema, incluindo o *script* de controle em Python e o banco vetorial ChromaDB, foi executada integralmente em uma estação de trabalho equipada com processador Intel Xeon E-2286G, 64 GB de RAM e GPU NVIDIA RTX A2000 (12 GB) sob Windows 10 Pro. Todos os modelos foram submetidos ao mesmo *prompt* de sistema (*Digital Guardian*) e avaliados pelas métricas de acurácia, precisão, *recall*, F1-score e tempo médio de resposta. Ressalta-se que, para os modelos em nuvem, o tempo de inferência mensurado inclui inevitavelmente a latência de rede (RTT - *Round Trip Time*); para mitigar essa variável, a estação de trabalho foi conectada via fibra óptica estável durante todos os experimentos.

4.1. Estudo de Caso: Aplicação em Cenário Real (UECE)

Além dos *benchmarks* padronizados, realizou-se um estudo de caso prático na Universidade Estadual do Ceará (UECE) para validar a detecção de engenharia social no contexto da Universidade[6].

4.1.1. Ambiente e Dinâmica

Diferentemente dos testes anteriores descritos na Seção 4, o banco vetorial ChromaDB foi pré-populado com 2.000 vetores provenientes dos datasets SMS e Enron, simulando um sistema em produção com conhecimento prévio. O fluxo de dados consistiu em mensagens submetidas voluntariamente por usuários reais durante uma onda de ataques de *phishing*.

Um componente central foi a ativação do *Active Learning* em tempo real, onde as interações dos usuários, confirmando ou corrigindo a resposta da IA, foram imediatamente vetorizadas. Isso permitiu que o sistema aprendesse dinamicamente termos específicos da instituição (como ”Degep”ou ”Cearaprev”), aprimorando o modelo para a captação de novas mensagens.

5. Resultados

Esta seção apresenta os resultados obtidos a partir da avaliação dos modelos de linguagem na classificação de mensagens de *e-mail* (Tabela 1) e SMS (Tabela 2). Foram avaliados modelos locais e baseados em nuvem, permitindo uma análise comparativa entre diferentes abordagens. Os resultados evidenciam diferenças significativas entre os modelos, tanto em termos de desempenho preditivo quanto de custo computacional.

[6]Disponibilidade dos Dados: Em conformidade com a LGPD, o conjunto bruto contendo as mensagens enviadas pelos usuários não será público. Entretanto, exemplos anonimizados estão disponíveis no repositório do projeto para fins de reprodutibilidade.

Tabela 1. Desempenho dos modelos na classificação de mensagens de e-mail

Modelo	Acur. (%)	Precisão (%)	Recall (%)	F1-Score (%)	Tempo (s)
llama3.2 (3B)	66,83	56,88	70,00	62,76	2,01
Llama 3 (8B)	68,14	56,74	86,71	68,59	**1,89**
Gemma 3 (12B)	71,34	59,19	92,03	72,04	15,64
Gemini 2.5 Flash	89,90	**100,00**	72,97	84,38	4,38
Gemini 3 Flash Preview	**97,73**	**100,00**	**94,44**	**97,14**	8,08
DeepSeek-v3.2	88,03	95,24	60,61	74,07	15,45

Tabela 2. Desempenho dos modelos na classificação de mensagens SMS.

Modelo	Acur. (%)	Precisão (%)	Recall (%)	F1-Score (%)	Tempo (s)
llama3.2 (3B)	70,95	31,50	73,05	44,02	1,52
Llama 3 (8B)	55,34	23,54	93,52	37,62	**1,28**
Gemma 3 (12B)	78,93	40,31	**96,30**	56,83	4,25
Gemini 2.5 Flash	89,90	70,59	70,59	70,59	2,30
Gemini 3 Flash Preview	**96,21**	**83,33**	95,24	**88,89**	7,11
DeepSeek-v3.2	89,20	59,62	83,78	69,66	13,56

Para a análise das Tabelas 1 e 2, as células destacadas em negrito indicam os melhores resultados.Comparando as duas tabelas, observa-se que todos os modelos apresentaram desempenho superior na classificação de e-mails quando comparados ao SMS, principalmente em relação à métrica F1-Score. Esse comportamento pode ser atribuído à maior quantidade de contexto presente em mensagens de e-mail, facilitando a identificação de padrões semânticos relevantes. Os resultados evidenciam um claro *trade-off* entre capacidade preditiva e custo computacional. Enquanto modelos robustos, como o Gemini 3 (Flash), lideraram em desempenho geral, modelos compactos executados localmente (como Llama 3B/8B) destacaram-se pela baixa latência, validando sua viabilidade em ambientes com recursos limitados.

No entanto, é importante ressaltar que os tempos de execução apresentados nas Tabelas 1 e 2 refletem exclusivamente a latência da etapa de inferência dos modelos generativos. O custo computacional da arquitetura RAG não está contabilizado nessas métricas.

As Tabelas 1 e 2 mostram que os modelos de menor porte, como Llama 3.2 (3B) e Llama 3 (8B), apresentaram os menores tempos de execução, indicando maior eficiência computacional. Em contrapartida, modelos mais robustos, como Gemma 3 (12B) e DeepSeek-v3.2, demandaram maior tempo de processamento, especialmente na classificação de mensagens de e-mail. De modo geral, o tempo médio de execução foi superior no cenário de e-mail quando comparado ao SMS, o que pode ser atribuído ao maior volume de texto e à complexidade semântica das mensagens analisadas.

5.1. Resultados do Estudo de Caso (UECE)

Neste cenário prático de alta complexidade, contendo gírias locais e URLs ofuscadas, o *VerificAI* alcançou uma taxa de acerto de 94% das mensagens submetidas durante o fluxo de validação. A análise qualitativa dos erros (6% restantes) revelou falhas majoritariamente em mensagens ambíguas que exigiam validação física externa.

Destaca-se um caso crítico interceptado que utilizava o tema de ”Prova de Vida”. O ataque empregava gatilhos de urgência e engenharia social visual, conduzindo a vítima a uma página falsa da universidade. A Figura 3 apresenta a mensagem original e a Figura 4

apresenta a análise gerada pela IA.

Assunto: [PROF-UECE] [COMUNICADO] Degep/Uece divulga lista de servidores que devem realizar, com urgência, Recadastramento e Prova de Vida

---

O Departamento de Gestão de Pessoas da Universidade Estadual do Ceará (Uece) reforça a necessidade de que os servidores realizem o Recadastramento e a Prova de Vida obrigatórios junto ao Estado do Ceará, a fim de evitar o bloqueio de seus vencimentos.
Segundo a Fundação de Previdência Social do Estado do Ceará (Cearaprev), os servidores estaduais ainda podem efetuar o Recadastramento e a Prova de Vida 2025.
Os processos estão disponíveis no aplicativo Cearaprev On-line. Utilize o link para concluir seu novo cadastro.
Em situações excepcionais, o Recadastramento e a Prova de Vida poderão ser realizados presencialmente na sede da Cearaprev.
O Degep/Uece divulga a lista de servidores da Universidade que ainda não realizaram os procedimentos e que devem regularizar a situação com urgência: (confira aqui).
O Departamento também divulga a lista de servidores que já tiveram seus vencimentos bloqueados (confira aqui). Após a realização do Recadastramento, o desbloqueio ocorre em até três dias úteis.

–
Assessoria de Comunicação UECE
Avenida Silas Munguba, 1700 - Campus Itaperi
Acompanhe nossas redes sociais: @ueceoficial

Figura 3. Mensagem de *phishing* interceptada.

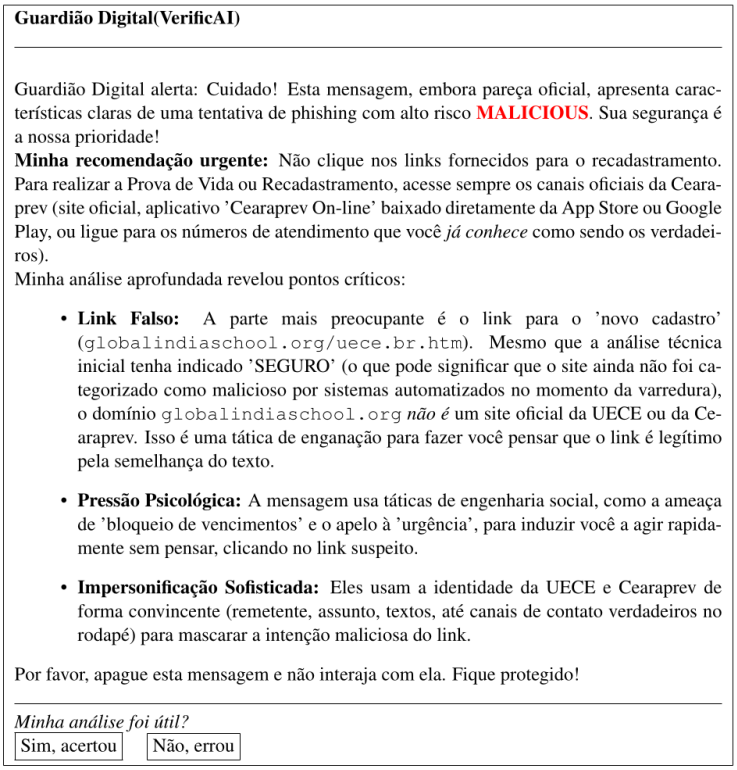
Como ilustrado na Figura 4, o *VerificAI* demonstrou sua eficiência em situações reais ao reconhecer corretamente o caráter malicioso da mensagem. A intervenção foi crucial, pois o ataque mostrado na Figura 3 representava um vetor ativo de engenharia social, criado para roubar as credenciais acadêmicas dos professores através de uma página de autenticação falsa.

6. Conclusão

Este trabalho avaliou a eficácia do *VerificAI* na detecção de ameaças baseadas em linguagem natural em mensagens SMS e *e-mails*, comparando diferentes modelos de linguagem, tamanhos de arquitetura e ambientes de execução. Os resultados demonstraram que modelos de grande porte executados em nuvem alcançam desempenho superior em métricas clássicas de classificação, como acurácia e F1-score, destacando-se na identificação de padrões complexos de *phishing*. Esses resultados destacam o potencial de arquiteturas modernas de LLMs, especialmente quando associadas a *prompts* padronizados e estratégias baseadas em RAG.

Por outro lado, os experimentos também evidenciaram que modelos menores (SLM) executados localmente apresentam tempos de inferência significativamente inferiores, configurando-se como alternativas viáveis em cenários que demandam baixa latência, maior controle sobre os dados e redução de custos computacionais. Essa análise revela um *trade-off* claro entre desempenho e eficiência, indicando que a escolha do modelo deve considerar os requisitos específicos do ambiente de implantação.

A aplicabilidade da proposta foi corroborada pela validação prática na Universidade Estadual do Ceará (UECE), onde o sistema demonstrou robustez frente a ataques de engenharia social contextualizados, classificando corretamente 94% das mensagens submetidas durante o fluxo de validação. Como trabalhos futuros, propõe-se a ampliação dos experimentos para bases de dados mais extensas e heterogêneas. Reconhece-se que a atual dependência da submissão manual de mensagens pelo usuário pode limitar a mitigação


Guardião Digital(VerificAI)

Guardião Digital alerta: Cuidado! Esta mensagem, embora pareça oficial, apresenta características claras de uma tentativa de phishing com alto risco **MALICIOUS**. Sua segurança é a nossa prioridade!

Minha recomendação urgente: Não clique nos links fornecidos para o recadastramento. Para realizar a Prova de Vida ou Recadastramento, acesse sempre os canais oficiais da Cearaprev (site oficial, aplicativo 'Cearaprev On-line' baixado diretamente da App Store ou Google Play, ou ligue para os números de atendimento que você *já conhece* como sendo os verdadeiros).

Minha análise aprofundada revelou pontos críticos:

- **Link Falso:** A parte mais preocupante é o link para o 'novo cadastro' (`globalindiaschool.org/uece.br.htm`). Mesmo que a análise técnica inicial tenha indicado 'SEGURO' (o que pode significar que o site ainda não foi categorizado como malicioso por sistemas automatizados no momento da varredura), o domínio `globalindiaschool.org` *não é* um site oficial da UECE ou da Cearaprev. Isso é uma tática de enganação para fazer você pensar que o link é legítimo pela semelhança do texto.
- **Pressão Psicológica:** A mensagem usa táticas de engenharia social, como a ameaça de 'bloqueio de vencimentos' e o apelo à 'urgência', para induzir você a agir rapidamente sem pensar, clicando no link suspeito.
- **Impersonificação Sofisticada:** Eles usam a identidade da UECE e Cearaprev de forma convincente (remetente, assunto, textos, até canais de contato verdadeiros no rodapé) para mascarar a intenção maliciosa do link.

Por favor, apague esta mensagem e não interaja com ela. Fique protegido!

Minha análise foi útil?
Sim, acertou | Não, errou

Figura 4. Análise e veredito do VerificAI.

eficaz das ameaças. Portanto, planeja-se a integração da arquitetura proposta diretamente em clientes de e-mail, navegadores (via extensões) e aplicativos de comunicação, operando de forma automatizada e transparente em segundo plano, de modo análogo a soluções antivírus. Além disso destaca-se o aprimoramento dos mecanismos de *Active Learning* para lidar com a evolução de longo prazo dos vetores de ataque, fortalecendo a aplicação do *VerificAI* em contextos reais de produção.

Agradecimentos

Pesquisa parcialmente financiada pelo CNPq (Processos N^o 305946/2025-0 e N^o 405940/2022-0) e Coordenação de Aperfeiçoamento de Pessoal de Nível Superior – Brasil (CAPES) – Código de Financiamento 88887.954253/2024-00.

Referências

Afonso, P., Maia, E., Amorim, I., and Praça, I. (2025). Rethinking phishing detection: How dataset quality affects model generalization. In *2025 15th International Conference on Advanced Computer Information Technologies (ACIT)*, pages 542–547.

Brito, M. L. L., Ferreira, M. C. M., Portela, A. L. C., and Gomes, R. L. (2026). Ai-based estimation of bandwidth availability for data offloading in edge-cloud computing. *IEEE Networking Letters*, 8:69–73.

Costa, M. A., Costa, Y. M., Almeida, Y. O., Cardoso, F. J., and Gomes, R. L. (2024). Connection management using automated firewall based on threat intelligence. In *Proceedings of the 2024 Latin America Networking Conference*, LANC '24, page 32–37, New York, NY, USA. Association for Computing Machinery.

Hasan, N., BusiReddyGari, P., Zhao, H., Ren, Y., Xu, J., and Zhang, S. (2025). Phishing email detection using large language models.

Koide, T., Fukushi, N., Nakano, H., and Chiba, D. (2024). Chatspamdetector: Leveraging large language models for effective phishing email detection.

Li, Z., Chen, W., and Zhang, H. (2025). Fedphishllm: A privacy-preserving and explainable phishing detection mechanism using federated learning and large language models. *Journal of Cybersecurity and Privacy*, 5(2):123–140.

Mahendru, S. and Pandit, T. (2024). Securenet: A comparative study of deberta and large language models for phishing detection. In *2024 IEEE 7th International Conference on Big Data and Artificial Intelligence (BDAI)*, page 160–169. IEEE.

Mendes, P., Maia, E., and Praça, I. (2025). Meajor corpus: A multi-source dataset for phishing email detection.

Pimenta, I., Silva, D., Moura, E., Silveira, M., and Gomes, R. L. (2024). Impact of data anonymization in machine learning models. In *Proceedings of the 13th Latin-American Symposium on Dependable and Secure Computing*, pages 188–191.

Pimenta, I. A., Lee, M. H., Bittencourt, L. F., and Gomes, R. L. (2025). Adaptive privacy based on mutual information for machine learning in edge-cloud environments. *IEEE Networking Letters*, pages 1–1.

Schmitt, M. and Flechais, I. (2024). Digital deception: generative artificial intelligence in social engineering and phishing. *Artificial Intelligence Review*, 57(12).

Souza, M. S., Ribeiro, S. E. S. B., Lima, V. C., Cardoso, F. J., and Gomes, R. L. (2024). Combining regular expressions and machine learning for sql injection detection in urban computing. *Journal of Internet Services and Applications*, 15(1):103–111.

Wang, Y., Tian, C., Hu, B., Yu, Y., Liu, Z., Zhang, Z., Zhou, J., Pang, L., and Wang, X. (2024). Can small language models be good reasoners for sequential recommendation? In *Proceedings of the ACM Web Conference 2024*, WWW '24, page 3876–3887, New York, NY, USA. Association for Computing Machinery.

Wang, Y., Zhai, H., Wang, C., Hao, Q., Cohen, N. A., Foulger, R., Handler, J. A., and Wang, G. (2025). Can you walk me through it? explainable sms phishing detection using llm-based agents. In *Proceedings of the Twenty-First USENIX Conference on Usable Privacy and Security*, SOUPS '25, USA. USENIX Association.

Xiao, S., Liu, Z., Zhang, P., Muennighoff, N., Lian, D., and Nie, J.-Y. (2024). C-pack: Packed resources for general chinese embeddings.

Xu, A., Yu, T., Du, M., Gundecha, P., Guo, Y., Zhu, X., Wang, M., Li, P., and Chen, X. (2024). Generative ai and retrieval-augmented generation (rag) systems for enterprise. In *Proceedings of the 33rd ACM International Conference on Information and Knowledge Management*, CIKM '24, page 5599–5602, New York, NY, USA. Association for Computing Machinery.