Guardrail: Uma Abordagem Modular para Sistemas de Segurança em Inteligência Artificial Generativa

Fábio Alves Bocampagni[1], **Daniel Sadoc Menasché**[1], **Renato Kogeyama**[2]

[1] Instituto de Computação – Universidade Federal do Rio de Janeiro (UFRJ)

[2]Kearney

{fabioab,sadoc}@ic.ufrj.br, renato.kogeyama@kearney.com

Abstract. *Guardrail is an open-source framework for protecting language models (LLMs) against threats such as prompt injection and jailbreaks. It follows a modular architecture inspired by the mixture of experts paradigm, combining lightweight and semantic filters. Evaluated on 1000 adversarial and legitimate prompts, Guardrail achieved an F1-score of 0.9844 (parallel mode) and 0.9711 (sequential), outperforming PromptGuard and LLM-Guard. These results highlight the benefits of combining specialized modules for robust LLM security.*

Resumo. *Guardrail é um framework open-source para proteger modelos de linguagem (LLMs) contra ataques como prompt injection e jailbreaks. Adota uma arquitetura modular inspirada em mixture of experts, combinando filtros leves e semânticos. Avaliado em 1000 prompts, obteve F1-score de 0,9844 (modo paralelo) e 0,9711 (sequencial), superando PromptGuard e LLM-Guard. Os resultados demonstram a eficácia da combinação de módulos especializados na segurança de LLMs.*

1. Introdução

O avanço dos LLMs trouxe à tona ameaças adversariais [Bick et al. 2024, TI Inside 2024]. Estudos recentes apontam que uma parcela significativa dos testes de *prompt injection* contra diversos modelos resulta em sucesso, revelando perfis de vulnerabilidade associados ao tamanho e à arquitetura das redes neurais modernas [The Alan Turing Institute 2024]. Além disso, ataques indiretos de *prompt injection*, que inserem instruções maliciosas em fontes de dados externas, podem passar despercebidos e comprometer pipelines de IA sem qualquer interação direta com o usuário, aumentando o risco de vazamentos e manipulação de informações [Piet et al. 2024, Das et al. 2025]. Esse cenário se torna ainda mais crítico considerando que 65% das empresas no mundo já utilizam IA generativa em seus processos [McKinsey & Company 2024].

Diante desse panorama de ameaças crescentes e da ampla adoção de modelos generativos, torna-se urgente o desenvolvimento de mecanismos de defesa que sejam ao mesmo tempo eficazes, escaláveis e adaptáveis a diferentes contextos de uso. Soluções atuais tendem a focar em abordagens monolíticas, com classificadores únicos treinados sobre grandes volumes de dados, o que pode limitar sua capacidade de generalização frente a ataques não vistos previamente ou comprometer sua aplicabilidade em cenários com restrições operacionais. Neste trabalho, propomos uma alternativa que explora a

composição de módulos especializados, permitindo combinar diferentes estratégias de detecção e mitigação de forma coordenada, com menor custo computacional e maior flexibilidade de adaptação [Liu et al. 2024, Greshake et al. 2023, Zhang et al. 2025].

Contribuições. As principais contribuições deste artigo são: (i) o desenvolvimento do *Guardrail*, um framework modular e *open-source* para defesa contra ataques de *prompt injection* em LLMs; (ii) a proposição de uma arquitetura inspirada em *mixture of experts*, permitindo a composição flexível de múltiplos filtros especializados — como análise de entropia, similaridade semântica, detecção de anomalias e validação com LLMs; (iii) a demonstração empírica, com benchmarks públicos, de que a abordagem híbrida do *Guardrail* supera métodos consolidados da indústria como *PromptGuard* e *LLM-Guard* em métricas como revocação e *F1-score*; e (iv) a validação da contribuição individual de cada módulo via teste de ablação, evidenciando os ganhos proporcionados pela modularidade e combinação de especialistas.

2. Trabalhos Relacionados

Os riscos de segurança associados ao uso de LLMs têm despertado crescente atenção na academia e na indústria. Dentre as ferramentas para mitigar ataques como *prompt injection*, uma das que se destaca é o *LLM Guard*[1], desenvolvida pela Protect AI, que atua como um *scanner* de segurança capaz de detectar e neutralizar comandos maliciosos. A solução combina múltiplas técnicas, incluindo sanitização de entrada, detecção de linguagem prejudicial e mecanismos para prevenção de vazamento de informações sensíveis.

Complementarmente, a Meta propôs o *PromptGuard-86M*[2], um classificador leve treinado em um amplo conjunto de ataques sintéticos e reais. O modelo busca identificar *prompts* adversariais com base em padrões previamente observados, reforçando a importância de abordagens multicamada na proteção de aplicações baseadas em LLMs.

No meio acadêmico, esforços recentes têm se concentrado na caracterização dos vetores de ataque mais sutis, como as injeções indiretas e os comandos contextuais encobertos [Zhou et al. 2023]. Suo et al. [Suo et al. 2024] introduzem o conceito de *Signed-Prompt*, no qual instruções sensíveis são criptograficamente assinadas por usuários autorizados, permitindo que o LLM diferencie comandos confiáveis de entradas manipuladas.

Por outro lado, [Sechel 2024] propõe o uso de técnicas de *embedding similarity* como mecanismo de defesa. O método compara vetores densos de *prompts* suspeitos com representações vetoriais de ataques conhecidos. Essa abordagem permite detectar variantes reescritas de comandos maliciosos, mesmo quando formuladas com vocabulário alternativo, ao estabelecer um limiar de confiança baseado em similaridade semântica.

Outra iniciativa relevante é o *NeMo Guardrails* [Rebedea et al. 2023], framework da NVIDIA voltado à proteção de aplicações baseadas em LLMs contra *jailbreaks* e *prompt injections*. O sistema combina políticas comportamentais programáveis — definidas em uma linguagem declarativa chamada `Colang` — com módulos de monitoramento e intercepção de fluxos de diálogo. Essa abordagem permite especificar restrições semânticas e lógicas com alto grau de auditabilidade, atuando preventivamente contra comportamentos indesejados. Embora o NeMo se diferencie por priorizar alinhamento e

[1] `https://llm-guard.com/`

[2] `https://huggingface.co/meta-llama/Prompt-Guard-86M`

Tabela 1. Comparação entre abordagens de defesa contra *prompt injection*

Atributo	LLM Guard	PromptGuard	Suo et al.	Sechel	NeMo	Guardrail
A1 - Similaridade semântica	✗	✗	✗	✓	✗	✓
A2 - Generalização / Zero-day	✓	✓	✓	✓	✓	✓
A3 - Uso de LLM para classif.	✓	✓	✗	✗	✓	✓
A4 - Modularidade	✓	✗	✗	✗	✓	✓
A5 - Mixture of Experts	✗	✗	✗	✗	✗	✓

conformidade de respostas em fluxos interativos, sua arquitetura orientada a regras pode ser integrada a pipelines de validação como o *Guardrail*.

A Tabela 1 contrasta as principais características dessas abordagens com a proposta deste trabalho. Cinco atributos são considerados, sendo que o *Guardrail* distingue-se por combinar todos os atributos em uma arquitetura única e extensível.

3. Metodologia

O projeto do *Guardrail* foi guiado por três princípios fundamentais: (i) **simplicidade antes da complexidade**, ou seja, iniciar a análise com heurísticas leves e de alta confiança sempre que possível, evitando o uso prematuro de recursos computacionais intensivos; (ii) **composição modular de especialistas**, permitindo que múltiplos filtros especializados sejam integrados de forma independente e coordenada; e (iii) **resiliência por diversidade**, com a combinação de estratégias complementares (estatísticas, semânticas e baseadas em LLMs) para mitigar diferentes vetores de ataque.

A arquitetura segue, por padrão, o *método sequencial* ilustrado na Figura 1. Cada entrada percorre filtros independentes. Caso um módulo inicial seja capaz de detectar, com alta confiança, a presença de um ataque, a execução é interrompida imediatamente, economizando recursos — alinhando-se ao princípio da simplicidade antes da complexidade. Essa abordagem é especialmente útil em ambientes com alta demanda ou restrições de latência.

Sequencial vs. Paralelo. O *Guardrail* oferece duas estratégias principais de execução: a abordagem sequencial — adotada como padrão — e a abordagem paralela, baseada no paradigma de *mixture of experts*. A escolha entre essas duas configurações reflete um trade-off central entre os princípios orientadores do sistema.

Na execução sequencial, aplicamos o princípio da **simplicidade primeiro**, priorizando filtros leves e de alta confiança logo no início do pipeline. Isso permite decisões rápidas e eficientes, economizando recursos sempre que um ataque é identificado precocemente. Essa configuração é particularmente vantajosa em ambientes com restrições de latência ou carga elevada.

Já na execução paralela, abrimos mão desse princípio ao aplicar todos os módulos simultaneamente e combinar suas respostas por meio de agregação. Embora essa abordagem envolva maior custo computacional, ela alcança desempenho levemente superior — como demonstrado na Tabela 2, em que o modo paralelo apresentou o maior *F1-score* (0,9844) entre todas as configurações testadas.

Assim, a escolha entre execução sequencial ou paralela depende diretamente das exigências operacionais e do perfil de risco da aplicação, permitindo que o Guardrail se

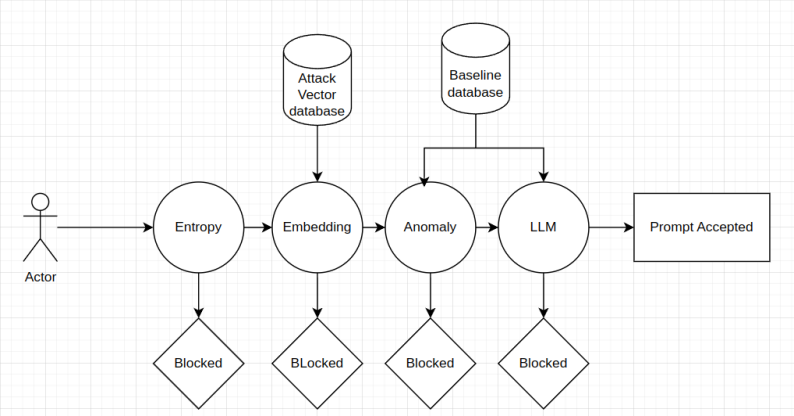

Figura 1. Configuração de validação sequencial

adapte de forma flexível a diferentes contextos de uso.

Dados. A construção dos módulos foi feita com base em dois conjuntos de dados distintos: um contendo **entradas legítimas**, usado como referência para os módulos de análise estatística e detecção de anomalias; e outro com **entradas maliciosas conhecidas**, usado para alimentar o módulo de similaridade semântica. Essa separação permite o uso de diferentes estratégias conforme o tipo de ameaça: desvios comportamentais em relação ao padrão legítimo ou proximidade semântica com ataques registrados.

Análise de Entropia. Este é o primeiro filtro ativado no pipeline, refletindo o princípio de começar com mecanismos leves. O módulo mede a imprevisibilidade da sequência textual enviada pelo usuário, com base na distribuição de tokens extraídos via NLTK. Entradas maliciosas tendem a apresentar características estatísticas fora do padrão, como excesso de repetições, uso incomum de símbolos ou estruturas gramaticais anômalas. A entropia calculada é comparada com uma média extraída de dados legítimos, permitindo detectar rapidamente outliers com alta confiança.

Similaridade Semântica por Embeddings. Este módulo detecta comandos semanticamente próximos de ataques previamente registrados, mesmo que reescritos de forma criativa. Utiliza o modelo *all-MiniLM-L6-v2*[3] para projetar o texto em um espaço vetorial denso. O vetor resultante é comparado a uma base de vetores de ataques conhecidos, previamente curada. Inicialmente hospedada no MongoDB Atlas[4], essa base foi posteriormente migrada para o *Faiss*[5], otimizando buscas de similaridade.

Detecção de Anomalias. Esse módulo visa identificar padrões atípicos em *prompts* aparentemente legítimos. Ao contrário da verificação por similaridade, que depende de histórico de ataques, aqui o foco está em desvios em relação ao comportamento típico. Para isso, foi treinado um classificador *SVM* com base em exemplos positivos (legítimos), considerando atributos como comprimento da entrada, variação lexical, entropia e uso de caracteres especiais. Isso permite detectar casos novos que não foram previamente catalogados.

[3]`https://huggingface.co/sentence-transformers/all-MiniLM-L6-v2`
[4]`https://www.mongodb.com/atlas`
[5]`https://github.com/facebookresearch/faiss`

Tabela 2. Desempenho dos frameworks para classificação de *prompt injection*

Modelo	Acurácia	Precisão	Revocação	F1-Score
Guardrail (Paralelo)	0.9750	0.9728	0.9962	0.9844
Guardrail (Sequencial)	0.9530	0.9450	0.9987	0.9711
Prompt-Guard (Meta)	0.7800	0.7887	0.9861	0.8764
LLM-Guard	0.7910	0.7911	1.0000	0.8833

Validação com Modelo de Linguagem. O último estágio do pipeline emprega uma LLM especializada na detecção de *prompt injection*. Utiliza-se um modelo *DeBERTa-v3 fine tuned*[6] para classificar a entrada como legítima ou adversarial. Essa etapa é computacionalmente mais custosa, mas é usada apenas como último recurso, garantindo alta precisão quando os módulos anteriores não fornecem uma resposta conclusiva. As saídas deste módulo podem alimentar os demais filtros em ciclos futuros de retreinamento, notando que a Fig. 1 corresponde a uma sequência única de inferência.

4. Resultados

Para avaliar a eficácia do *Guardrail*, conduzimos um experimento controlado utilizando um conjunto de 1000 entradas extraídas da base de dados proposta por [Sharma et al. 2024], composta por exemplos reais e sintéticos de *prompt injection*, além de instruções legítimas. A diversidade semântica e estrutural desse conjunto torna-o adequado para testar a robustez de sistemas de defesa em contextos realistas e variados.

A Tabela 2 apresenta os resultados obtidos por quatro abordagens: duas variantes do Guardrail (modo paralelo, via *mixture of experts*, e sequencial), o PromptGuard (Meta) e o LLM-Guard (Protect AI). Foram avaliadas as métricas de acurácia, precisão, revocação e F1-score, visando analisar o efeito de falsos positivos e falsos negativos.

Os resultados mostram que o *Guardrail* supera consistentemente as abordagens concorrentes em todas as métricas. No modo *mixture of experts* (MoE), o sistema obteve o melhor *F1-score* (0,9844), combinando alta revocação com precisão robusta — essencial para cenários em que a detecção de ataques deve ser maximizada sem comprometer a usabilidade do sistema. A variante sequencial também apresentou desempenho competitivo, com *F1-score* de 0,9711 e a maior revocação registrada (0,9987), evidenciando a eficácia do princípio da avaliação incremental com custos reduzidos.

Em contraste, soluções como o PromptGuard e o LLM-Guard priorizam revocação em detrimento da precisão. Embora ambos consigam identificar a maioria dos ataques (revocação de 0,9861 e 1,000, respectivamente), seus *F1-scores* indicam maior propensão a falsos positivos, o que pode comprometer a experiência do usuário em aplicações reais.

A Figura 2 ilustra o desempenho do *Guardrail* no modo paralelo (*mixture of experts*), revelando uma estabilidade notável ao longo do conjunto de testes, com alta taxa de acertos mesmo diante de entradas adversariais com variações sutis. Já a Figura 3 apresenta o comportamento no modo sequencial, que mantém desempenho competitivo mesmo com custo computacional reduzido, graças à interrupção antecipada de execução em casos de alta confiança. Essa configuração mostra-se especialmente eficaz em cenários com restrições operacionais, como baixa latência ou limitação de recursos.

[6] `https://huggingface.co/protectai/deberta-v3-base-prompt-injection`

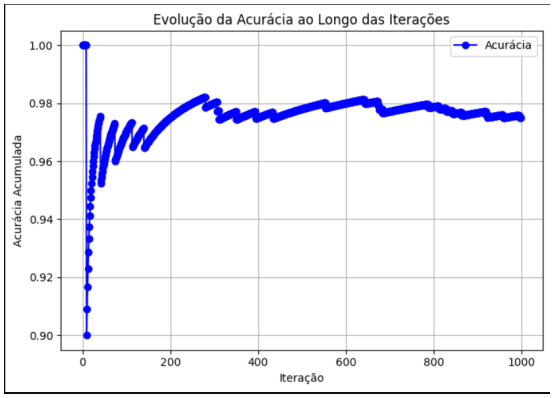

Figura 2. Execução paralela

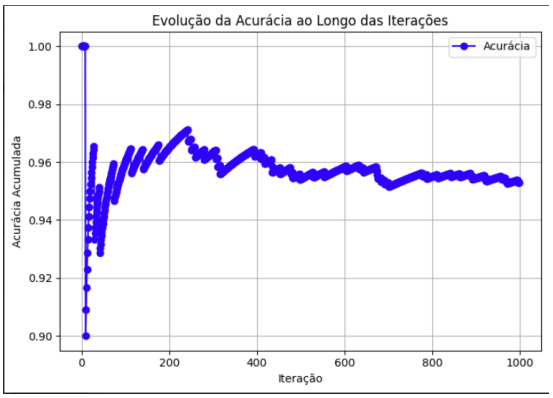

Figura 3. Execução sequencial

A Figura 4 complementa a análise ao apresentar os resultados do *Ablation Test*, no qual os módulos do sistema foram avaliados individualmente e em combinações parciais. Os resultados indicam que, embora alguns módulos — como a validação via LLM e o filtro de similaridade semântica — possuam desempenho razoável de forma isolada, nenhum deles é suficiente para alcançar os níveis de revocação e precisão obtidos pela configuração completa. Essa observação reforça a hipótese central deste trabalho: a superioridade do *Guardrail* decorre da interação entre múltiplos filtros especializados, que juntos cobrem diferentes vetores de ataque de forma complementar e eficaz.

A Tabela 3 apresenta os tempos médios de resposta dos diferentes modos de execução e módulos que compõem o *Guardrail*. No modo paralelo, implementado via *Mixture of Experts* (MoE), a execução completa — com todos os modelos já em memória — apresenta tempo médio de 45,90 milissegundos, sendo adequada para cenários onde se privilegia máxima cobertura e robustez frente a ataques sutis. Em contrapartida, o modo sequencial permite ganhos expressivos de desempenho ao explorar a execução condicional dos módulos. Por exemplo, os tempos de resposta dos módulos de detecção de anomalia (4,76 ms) e entropia (2,32 ms) são significativamente inferiores e, em casos de alta confiança, podem interromper a cadeia de processamento antes da validação final com LLM (12,8 ms), reduzindo o custo computacional total. O módulo de similaridade semântica, frequentemente acionado no início da execução, apresenta tempo médio de 18,3 ms. Esses resultados demonstram que o modo sequencial pode, em determinadas situações, superar o MoE em termos de eficiência, sem comprometer a precisão. Todos os valores reportados foram obtidos por meio de experimentos descritos no arquivo `benchmark.py`, disponível no repositório do projeto.

Os testes foram realizados em uma máquina com Ubuntu 24.04.2 LTS, equipada com um processador AMD Ryzen 5 5600G (12 threads), 32 GB de memória RAM e 2,5 TB de armazenamento. O sistema também conta com uma GPU dedicada NVIDIA GeForce RTX 3060 Ti, utilizada nos módulos compatíveis com aceleração via CUDA, como a validação com LLM e a vetorização por embeddings. Os demais módulos foram executados na CPU. Todos os testes consideraram os modelos previamente carregados em memória, isolando os tempos de inferência dos tempos de inicialização.

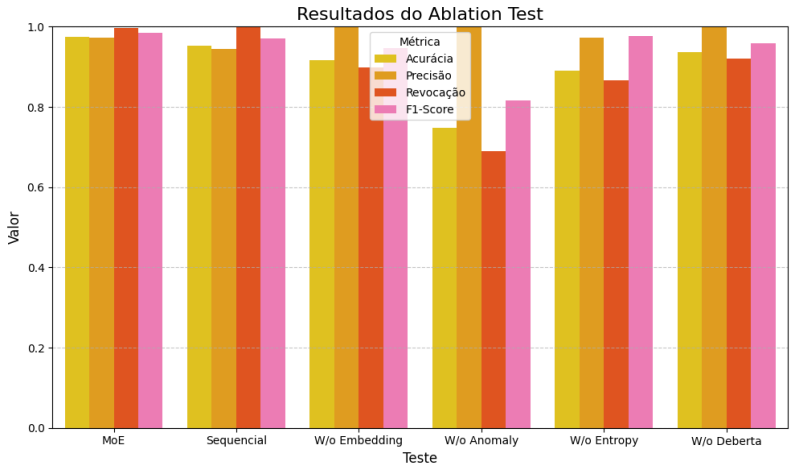

Figura 4. *Ablation Test*

Tabela 3. Tempos de resposta médios dos modos de execução e módulos do *Guardrail*

Modo / Módulo	**Descrição**	**Tempo médio (ms)**
Execução no modo *Mixture of Experts* (MoE)		
MoE	Execução padrão com modelos já em memória	45.90
Execução no modo sequencial (por módulo)		
Similaridade semântica (embedding)	Busca e comparação vetorial usando embeddings	18.3
Detecção de anomalia (SVM)	Classificação com base em padrões aprendidos	4.76
Entropia	Análise estatística da entrada textual	2.32
Validação com LLM	Classificação final usando DeBERTa-v3	12.8

5. Conclusão

Arquiteturas modulares baseadas na composição de filtros especializados podem oferecer uma defesa eficaz e adaptável contra *prompt injection* em modelos de linguagem. O *Guardrail*, ao integrar heurísticas estatísticas, verificação semântica e validação com LLMs, obteve desempenho superior às soluções existentes, combinando alta revocação com baixo custo computacional. Essa abordagem se destaca especialmente em cenários com restrições operacionais, onde a eficiência e a flexibilidade são requisitos.

Apesar dos resultados promissores, a segurança de LLMs segue como um desafio em constante evolução. Novos vetores, como injeções multi-turn e comandos contextuais disfarçados, exigem defesas adaptáveis e em contínuo aperfeiçoamento. Como próximos passos, propomos adaptar o Guardrail a domínios sensíveis — como saúde, direito e finanças — e explorar modelos mais expressivos para detectar padrões adversariais complexos. A modularidade da arquitetura facilita essas extensões.

Artefatos. Todos os artefatos mencionados estão disponíveis de forma pública em `https://github.com/Bocampagni/guardrail`.

Agradecimentos. Este projeto foi financiado em parte pela CAPES, CNPq e FAPERJ (E-26/204.268/2024).

Referências

Bick, A., Blandin, A., and Deming, D. J. (2024). The Rapid Adoption of Generative AI. Technical report, Federal Reserve Bank of St. Louis.

Das, B. C., Amini, M. H., and Wu, Y. (2025). Security and privacy challenges of large language models: A survey. *ACM Computing Surveys*, 57(6):1–39.

Greshake, K., Abdelnabi, S., Mishra, S., Endres, C., Holz, T., and Fritz, M. (2023). Not what you've signed up for: Compromising real-world llm-integrated applications with indirect prompt injection. In *Proceedings of the 16th ACM Workshop on Artificial Intelligence and Security*, pages 79–90.

Liu, Y., Jia, Y., Geng, R., Jia, J., and Gong, N. Z. (2024). Formalizing and benchmarking prompt injection attacks and defenses. In *33rd USENIX Security Symposium (USENIX Security 24)*, pages 1831–1847.

McKinsey & Company (2024). 65% das empresas usam Gen AI no mundo. https://www.mckinsey.com.br/our-insights/all-insights/65-das-empresas-usam-gen-ai-no-mundo. Accessed: 2025-04-10.

Piet, J., Alrashed, M., Sitawarin, C., Chen, S., Wei, Z., Sun, E., Alomair, B., and Wagner, D. (2024). Jatmo: Prompt injection defense by task-specific finetuning. In *European Symposium on Research in Computer Security*, pages 105–124. Springer.

Rebedea, T., Dinu, R., Sreedhar, M., Parisien, C., and Cohen, J. (2023). NeMo Guardrails: A toolkit for controllable and safe LLM applications with programmable rails. *arXiv preprint arXiv:2310.10501*.

Sechel, S. (2024). Adversarial testing and prompt injection attacks using the embedding similarity approximation method.

Sharma, R. K., Gupta, V., and Grossman, D. (2024). SPML: A DSL for Defending Language Models Against Prompt Attacks.

Suo, X. et al. (2024). Signed-prompt: A new approach to prevent prompt injection attacks against llm-integrated applications. *arXiv preprint*.

The Alan Turing Institute (2024). Indirect Prompt Injection: Generative AI's Greatest Security Flaw. https://cetas.turing.ac.uk/publications/indirect-prompt-injection-generative-ais-greatest-security-flaw. Accessed: 2025-04-10.

TI Inside (2024). IA generativa se tornará uma indústria de US$ 100 bilhões até 2026. https://tiinside.com.br/29/01/2024/ia-generativa-se-tornara-uma-industria-de-us-100-bilhoes-ate-2026. Accessed: 2025-04-10.

Zhang, X., Zhang, C., Li, T., Huang, Y., Jia, X., Hu, M., Zhang, J., Liu, Y., Ma, S., and Shen, C. (2025). JailGuard: A Universal Detection Framework for Prompt-based Attacks on LLM Systems. *ACM Transactions on Software Engineering and Methodology*.

Zhou, Y. et al. (2023). Prompt Injection Attack against LLM-integrated Applications. *arXiv preprint arXiv:2306.05499*. https://arxiv.org/abs/2306.05499.