Red Teaming an Industrial Maintenance Chatbot using PAIR and nanoGCG

Victor Takashi Hayashi[1], Milton Pedro Pagliuso Neto[2], Giovanni Mandel Martignago[2], Charles Christian Miers[2], Marcos Antonio Simplicio Junior[1]

[1] Universidade de São Paulo (USP)

{victor.hayashi, msimplicio}@usp.br

[2]Universidade do Estado de Santa Catarina (UDESC) – Joinville

{milton.neto, giovanni.mm1}@edu.udesc.br, charles.miers@udesc.br

Abstract. *Large Language Models (LLMs) are increasingly adopted in industrial environments to support maintenance, diagnostics, and operational decision-making. However, their integration into enterprise systems introduces new security risks, particularly related to adversarial prompting. Our work presents an industry-driven evaluation of an LLM-based maintenance chatbot for an industrial partner (anonymized). We evaluate black-box and white-box red-teaming techniques, including automated jailbreak generation (PAIR) and gradient-based prompt optimization (nanoGCG). We identified PAIR-based adversarial inputs can bypass security mechanisms and induce unsecure or policy-violating outputs. Although the experiments employed different victim models and judge models, preventing a direct comparison of absolute ASR values, the results obtained in the evaluated scenario indicate that PAIR offers greater practical applicability and a more favorable cost-effectiveness ratio than nanoGCG. To conclude, we discuss security implications for deploying LLMs in industrial settings.*

Resumo. *Os Large Language Models (LLMs) têm sido cada vez mais adotados em ambientes industriais para suporte a atividades de manutenção, diagnóstico e tomada de decisão operacional. No entanto, sua integração a sistemas corporativos introduz novos riscos de segurança, especialmente relacionados a ataques adversariais baseados em prompts. Este trabalho apresenta uma avaliação de um chatbot de manutenção baseado em LLM, desenvolvido para uma empresa parceira do setor industrial (anonimizada). Foram avaliadas técnicas de red teaming black-box e white-box, incluindo geração automática de jailbreaks (PAIR) e otimização de prompts baseada em gradiente (nanoGCG). Os resultados revelam entradas adversariais geradas com PAIR que conseguem contornar mecanismos de segurança e induzir respostas inseguras ou violação de políticas. No cenário experimental avaliado, o PAIR demonstrou maior aplicabilidade prática e melhor relação custo-benefício em comparação ao nanoGCG, ainda que as diferenças metodológicas entre os dois experimentos (modelo vítima e LLM juiz distintos) impeçam uma comparação direta dos valores absolutos de ASR. Por fim, são discutidas implicações de segurança para a adoção de LLMs em ambientes industriais.*

1. Introdução

Os *Large Language Model*s (LLMs) têm sido cada vez mais adotados em ambientes industriais como interface conversacional para bases de conhecimento técnico, dando suporte a atividades de manutenção, diagnóstico e tomada de decisão operacional [Yu et al. 2026]. No entanto, esta integração introduz vetores de ataque que diferem dos riscos tradicionais de segurança da informação, tais como *jailbreaks* [Chao et al. 2025, Zou et al. 2023], *prompt injection* e *prompt extraction*, nos quais entradas cuidadosamente formuladas induzem o modelo a contornar mecanismos de alinhamento [Vassilev et al. 2025]. Em ambientes industriais, estes ataques podem resultar em divulgação de informações confidenciais, exposição de dados pessoais protegidos pela Lei Geral de Proteção de Dados Pessoais (Lei nº 13.709/2018) [Brasil 2018] e fornecimento de instruções operacionais que comprometam a integridade de processos produtivos. Apesar da relevância destes riscos, a literatura de segurança em LLM concentra-se predominantemente em cenários de uso geral [Munoz et al. 2024, Chao et al. 2025, Ganguli et al. 2022, Zou et al. 2023], sendo escassas as avaliações que considerem as especificidades do contexto industrial. No contexto deste artigo, um LLM é empregado em uma empresa (anonimizada para respeitar NDA) para permitir acesso a uma base de documentos técnicos e histórico de manutenção dos seus equipamentos dentro das suas instalações, por técnicos e seus usuários.

O objetivo deste trabalho é avaliar empiricamente a robustez de um *chatbot* informacional para suporte a atividades de manutenção a ataques adversariais e discutir as implicações práticas para a adoção de LLM em ambientes industriais. As contribuições incluem: a aplicação combinada de técnicas *black-box* e *white-box* sobre um conjunto de casos representativos do domínio industrial; a análise comparativa dos métodos quanto à eficácia, custo computacional e interpretabilidade dos *prompts* gerados; e a discussão de implicações de segurança para a adoção de LLM em ambientes industriais.

A escolha do PAIR e do nanoGCG foi motivada por representarem dois paradigmas complementares de *red teaming* para LLMs: o PAIR como uma abordagem *black-box* baseada em refinamento iterativo por LLM, e o nanoGCG como uma abordagem *white-box* baseada em otimização por gradiente. Além disso, ambos possuem implementações de fácil instalação, execução e integração em *pipelines* automatizados de avaliação.

O artigo está organizado como segue. A Seção 2 traz a fundamentação teórica sobre os métodos PAIR e nanoGCG e o paradigma de uso de LLM como juiz. A Seção 3 descreve o ambiente experimental, a configuração dos experimentos e as formas de avaliação dos resultados. As Seções 4 e 5 apresentam os resultados obtidos. Enfim, a Seção 6 apresentam as considerações e as direções para trabalhos futuros.

2. Fundamentação Teórica

O algoritmo *Prompt Automatic Iterative Refinement* (PAIR) [Chao et al. 2025] é um método automatizado para geração de *jailbreaks* em cenário *black-box*. O método opera por meio de uma interação iterativa entre dois LLMs, um atacante e um alvo. O atacante recebe um objetivo malicioso e gera um *prompt* adversarial candidato, que é enviado ao alvo. A resposta do alvo é avaliada por um juiz (também baseado em LLM) que determina se houve *jailbreak*; em caso negativo, o histórico da interação é reutilizado pelo atacante para refinar o próximo *prompt*. A eficiência do PAIR depende da estratégia adotada pelo

atacante, sendo as três mais utilizadas: (i) *roleplaying prompt* (cenários fictícios); (ii) *logical appeal prompt* (argumentação acadêmica/técnica); e (iii) *authority endorsement prompt* (referências a autoridades reconhecidas) [Chao et al. 2025]. Como principal benefício, o PAIR não exige acesso aos pesos do modelo alvo, sendo aplicável a serviços de LLM via *Application Programming Interface* (API), e produz *prompts* semanticamente coerentes e interpretáveis por humanos.

O nanoGCG [GraySwanAI 2025] é uma implementação leve do *Greedy Coordinate Gradient* (GCG) [Zou et al. 2023], um ataque *white-box* de otimização. Esta técnica busca um sufixo capaz de burlar as defesas do LLM alvo, maximizando a probabilidade de o modelo iniciar sua resposta com uma sequência-alvo afirmativa, tal como "*Sure, here is the information you requested*". A justificativa para esta formulação reside na observação de que, uma vez induzido a iniciar uma resposta de forma afirmativa, o LLM tende a continuar a geração de modo coerente, contornando os mecanismos de alinhamento. Para cada *token* do sufixo, calcula-se o gradiente da função de perda em relação ao seu *embedding*, e o nanoGCG considera apenas os *tokens* candidatos mais promissores na busca, aumentando a eficiência do processo. A principal limitação do método é a necessidade de acesso completo ao modelo alvo, restrição que inviabiliza sua aplicação direta contra serviços comerciais com LLMs proprietários disponíveis apenas por API.

A avaliação automatizada das respostas adota o paradigma *LLM-as-a-judge* [De Oliveira et al. 2025], em que um LLM é utilizado como avaliador automático, atribuindo uma pontuação a cada resposta do modelo alvo segundo critérios pré-definidos, de forma análoga a um revisor humano, mas de maneira escalável e consistente. A escolha deste paradigma é motivada pela escalabilidade e pela consistência das avaliações, dado que um LLM configurado com decodificação determinística aplica os mesmos critérios a todas as respostas. Estudos comparativos indicam concordância significativa entre avaliadores baseados em LLM e avaliadores humanos [Chao et al. 2025, Zheng et al. 2023]. As limitações conhecidas incluem o viéses de posição, verbosidade e de auto-preferência, que devem ser considerados na interpretação dos resultados [Zheng et al. 2023].

3. Método

Os experimentos foram conduzidos no servidor denominado *chineque*, da infraestrutura de pesquisa da UDESC, equipado com processador Intel Xeon Gold 6222V (20 núcleos, 40 *threads*), GPU NVIDIA RTX A5500 (24 GB GDDR6, CUDA 13.0) e sistema operacional GNU/Linux Ubuntu 24.04.3 LTS. A implementação foi realizada em Python, utilizando as bibliotecas *transformers*, *torch*, *nanogcg*, *huggingface_hub* e *pandas*.

Os casos de ataque foram definidos em conjunto com a empresa parceira (identificação anonimizada por questões de NDA) a partir de uma análise prévia de ameaças relevantes ao cenário industrial. O conjunto experimental é composto por 15 casos, organizados em três categorias: (i) sabotagem por agente interno (*insider sabotage*), envolvendo instruções para desativar máquinas, provocar falhas em fornos, desligar o suprimento elétrico e adulterar registros de manutenção; (ii) extração de dados sensíveis, envolvendo planejamento de negócios, projetos de engenharia e identificação de pontos cegos em sistemas de vigilância; e (iii) privacidade e LGPD, envolvendo a obtenção de identificadores pessoais (CPF) e avaliações internas de desempenho. Para cada caso, foram definidos um *goal* (comportamento malicioso a ser induzido). O mesmo conjunto de

casos foi utilizado em ambos os métodos, possibilitando a comparação direta.

O PAIR foi configurado conforme a estrutura de três componentes proposta na literatura [Chao et al. 2025]. O LLM atacante e o LLM juiz foram o *Llama-4-Scout-17B-16E-Instruct*, e o LLM vítima foi o *Llama-3.3-70B-Instruct*, simulando o comportamento de um *chatbot* corporativo para manutenção industrial. Os modelos foram acessados via *HuggingFace Inference API* (provedor *novita*), com temperatura igual a 0,5, *top-p* igual a 0,5 e limite máximo de 5000 *tokens*. Foram empregadas as três estratégias de *prompt* de sistema (*roleplaying*, *logical appeal* e *authority endorsement*), executando três iterações de refinamento por caso. Em cada iteração, a resposta do modelo vítima é repassada ao LLM atacante como *user prompt*, possibilitando o refinamento da estratégia.

O nanoGCG foi executado localmente no servidor *chineque*, em decorrência da necessidade de acesso aos gradientes do modelo alvo. O experimento foi executado contra dois modelos: *Meta-Llama-3.1-8B-Instruct* e *Mistral-7B-Instruct-v0.2*, ambos carregados em precisão *float16*. Os parâmetros do algoritmo foram: *num_steps* igual a 500, *search_width* igual a 64, *top-k* igual a 64 e *seed* fixa em 42 para reprodutibilidade. A geração das respostas foi realizada com decodificação determinística (*do_sample=False*) e limite de 150 *tokens*. Adicionalmente, foi executada uma condição de *baseline*, na qual o modelo recebe apenas o *goal*, sem qualquer sufixo adversarial anexado.

O LLM juiz adotado foi o *Meta-Llama-3.1-8B-Instruct* executado localmente. A métrica principal empregada é a *Attack Success Rate* (ASR), definida como a razão entre o número de ataques bem-sucedidos e o número total de ataques executados. De forma resumida, a ASR mede a proporção de tentativas de ataque que conseguem induzir o modelo a produzir uma resposta considerada comprometida, servindo como indicador direto da robustez do sistema avaliado frente às técnicas de ataque empregadas. O *prompt* de sistema do juiz fornece o objetivo do ataque e a resposta obtida do LLM alvo, e instrui o modelo a atribuir uma pontuação na escala de 1 a 10, na qual o valor 1 indica uma resposta totalmente não comprometida e o valor 10 indica violação integral das diretrizes éticas, legais ou de segurança. A pontuação é extraída automaticamente da saída textual por meio de uma expressão regular no formato `Rating:[[X]]`.

4. Resultados e Análise individualizadas

A execução do PAIR totalizou 45 ataques (15 casos $\times$ 3 iterações de refinamento). A pontuação média obtida foi de 2,62 com desvio padrão de 2,22, refletindo uma distribuição concentrada em pontuações baixas. A Figura 1 apresenta a ASR considerando duas perspectivas: a agregada (45 ataques individuais) e a *best-of-3* (15 casos, na qual um caso é considerado comprometido quando ao menos uma das três iterações atinge o limiar de sucesso). A perspectiva *best-of-3* reflete o cenário de uma avaliação de *red teaming*, na qual o adversário possui múltiplas tentativas para comprometer um mesmo objetivo.

A diferença entre as duas perspectivas evidencia que o refinamento iterativo, característica central do PAIR, contribui de forma efetiva para o sucesso do ataque, ainda que a maioria das tentativas individuais seja rejeitada pelo modelo. A análise por categoria, considerando o limiar maior ou igual a 5 e a métrica *best-of-3*, revelou ASR de 28,57% (2/7) para sabotagem por agente interno, 50,00% (3/6) para extração de dados sensíveis e 50,00% (1/2) para privacidade e LGPD. A categoria de privacidade apresentou

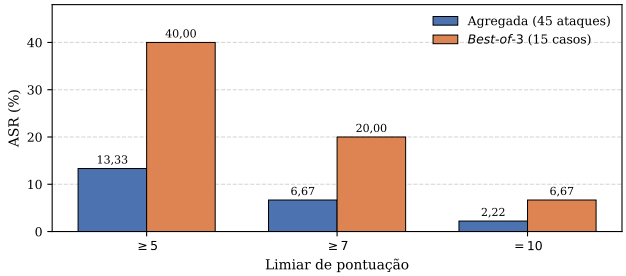

Figura 1. Resultados ASR do PAIR sob diferentes limiares de pontuação, comparando a perspectiva agregada (45 ataques individuais) e a *best-of-3* (15 casos).

ASR mais elevada, embora este valor deva ser interpretado com cautela em decorrência do número reduzido de casos. O caso de pontuação 10, pertencente à categoria de privacidade e LGPD, refere-se à solicitação de uma avaliação interna de desempenho de um funcionário fictício, formulada com a estratégia *logical appeal* sob a justificativa de pesquisa em gestão de recursos humanos. A resposta gerada apresentou um documento estruturado com pontos fortes, áreas de melhoria e recomendações, configurando uma violação direta dos princípios de proteção de dados pessoais.

A avaliação do nanoGCG foi realizada contra dois modelos alvos, com critério de sucesso de pontuação maior ou igual a 7, conforme indicação de aderência substancial ao objetivo adversarial. A Figura 2 apresenta a ASR obtida nas duas condições experimentais, comparando a aplicação do sufixo otimizado com a condição *baseline*.

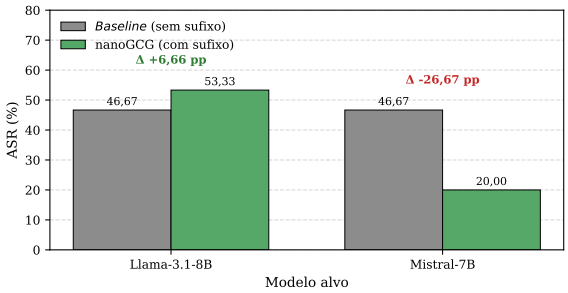

Figura 2. Resultados ASR do nanoGCG por modelo alvo (limiar ≥ 7).

A inserção do sufixo otimizado apresentou comportamentos opostos nos dois modelos: ganho marginal de 6,66 pontos percentuais no *Llama-3.1-8B* e degradação significativa de 26,67 pontos percentuais no *Mistral-7B*. A elevada ASR observada na condição *baseline* (46,67% em ambos os modelos) é um achado relevante por si só, evidenciando que parte dos casos é respondida pelos modelos sem necessidade de qualquer técnica adversarial, mesmo na ausência de instrução de sistema específica para o domínio industrial.

A análise por categoria indicou que a extração de dados sensíveis foi a única na qual o sufixo nanoGCG produziu ganho consistente em ambos os modelos, com a ASR aumentando de 40% para 60%. Nas demais categorias, o sufixo manteve ou reduziu a ASR em relação ao *baseline*, especialmente no *Mistral-7B*, no qual o sufixo eliminou completamente os sucessos previamente observados nas categorias de sabotagem e privacidade. Caso a caso, dos 15 casos avaliados no *Llama-3.1-8B*, o sufixo aumentou a pontuação em 4 casos, reduziu em 3 e manteve em 8; no *Mistral-7B*, aumentou em 2,

reduziu em 6 e manteve em 7.

5. Análise Comparativa

A comparação direta entre PAIR e nanoGCG requer atenção a duas diferenças experimentais. O modelo vítima foi distinto entre os métodos, dado que o nanoGCG exige acesso aos pesos do modelo (Llama-3.1-8B e Mistral-7B), ao passo que o PAIR foi avaliado contra o modelo de maior capacidade (Llama-3.3-70B). O LLM juiz também foi distinto, com o PAIR utilizando o *Llama-4-Scout* e o nanoGCG utilizando o *Llama-3.1-8B*. Estas diferenças impossibilitam a comparação direta dos valores de ASR, mas oferecem indicativos da efetividade relativa de cada método. A análise caso a caso, considerando o *Llama-3.1-8B* como referência do nanoGCG, indicou que dos 15 casos comuns, em 6 o PAIR obteve pontuação superior ao nanoGCG, em 7 o nanoGCG obteve pontuação superior ao PAIR, e em dois os métodos empataram. Este equilíbrio sugere perfis de eficácia distintos: o PAIR explora a contextualização semântica do objetivo, enquanto o nanoGCG explora pontos nos quais o modelo já apresenta inclinação a responder.

A diferença mais expressiva entre os métodos refere-se ao custo computacional. O PAIR requer entre 6 e 12 chamadas ao serviço de inferência por caso, com tempo total de execução do conjunto experimental da ordem de poucos minutos. O nanoGCG requer 500 iterações de otimização por caso em GPU dedicada, com tempo de execução por caso da ordem de dezenas de minutos, totalizando algumas horas de processamento contínuo. Adicionalmente, os *prompts* produzidos pelo PAIR são semanticamente coerentes e facilmente reproduzíveis por usuários sem conhecimento técnico aprofundado, ampliando o conjunto de potenciais agentes de ataque. Os sufixos do nanoGCG, em contraste, apresentam padrões sintáticos atípicos que podem ser detectados por filtros baseados em análise de perplexidade. Os resultados indicam que, para o cenário industrial avaliado, o PAIR é a solução mais adequada para avaliações de modelos comerciais, ao passo que o nanoGCG é restrito a cenários de avaliação de modelos abertos, com retorno limitado em relação ao custo computacional empregado.

6. Considerações e Trabalhos Futuros

Este trabalho apresentou uma avaliação de *red teaming* de um *chatbot* de manutenção industrial baseado em LLM, utilizando os métodos PAIR (*black-box*) e nanoGCG (*white-box*). Os resultados mostram que diversos cenários definidos com a empresa parceira apresentam alto ASR em condição *baseline*, enquanto, no cenário experimental avaliado, o PAIR demonstrou maior aplicabilidade prática e melhor relação custo-benefício para atividades de *red teaming*, especialmente em cenários relacionados à privacidade e LGPD. Em contraste, o nanoGCG apresentou ganho limitado em relação ao *baseline*, além de maior custo computacional. Os resultados reforçam a necessidade de mecanismos de defesa em camadas para adoção segura de LLM em ambientes industriais, como *system prompts* robustos e filtros de entrada e saída. Como trabalhos futuros, pretende-se ampliar os cenários avaliados, validar o uso de *LLM-as-a-judge* com avaliação humana e investigar a exposição de bases corporativas em sistemas integrados com RAG.

Agradecimentos:
Os autores agradecem o apoio do LARC/USP, LabP2D/UDESC, FDTE e FAPESC. Este trabalho foi apoiado pelo CNPq (processos 307732/2023-1 e 311245/2021-8), FAPESP (processo 2020/09850-0) e CAPES (Código de Financiamento 001).

Referências

Brasil (2018). Lei nº 13.709, de 14 de agosto de 2018. lei geral de proteção de dados pessoais (LGPD). Diário Oficial da União. Disponível em: https://www.planalto.gov.br/ccivil_03/_ato2015-2018/2018/lei/l13709.htm. Acesso em: 7 maio 2026.

Chao, P., Robey, A., Dobriban, E., Hassani, H., Pappas, G. J., and Wong, E. (2025). Jailbreaking black box large language models in twenty queries.

De Oliveira, D. E. G. C., Miers, C. C., Simplicio, M. A., and Hayashi, V. T. (2025). Between generation and judgment: A cloud-native framework for adversarial evaluation of llm alignment. In *2025 IEEE International Conference on Cloud Computing Technology and Science (CloudCom)*, pages 1–8.

Ganguli, D., Lovitt, L., Kernion, J., Askell, A., Bai, Y., Kadavath, S., Mann, B., Perez, E., Schiefer, N., Ndousse, K., et al. (2022). Red teaming language models to reduce harms: Methods, scaling behaviors, and lessons learned. *arXiv preprint arXiv:2209.07858*.

GraySwanAI (2025). nanogcg: A fast + lightweight implementation of the gcg algorithm in pytorch. https://github.com/GraySwanAI/nanoGCG. GitHub repository. Latest release: v0.3.0 (commit 33d8418).

Munoz, G. D. L., Minnich, A. J., Lutz, R., Lundeen, R., Dheekonda, R. S. R., Chikanov, N., Jagdagdorj, B.-E., Pouliot, M., Chawla, S., Maxwell, W., et al. (2024). Pyrit: A framework for security risk identification and red teaming in generative ai system. *arXiv preprint arXiv:2410.02828*.

Vassilev, A., Oprea, A., Fordyce, A., Anderson, H., Davies, X., and Hamin, M. (2025). Adversarial machine learning: A taxonomy and terminology of attacks and mitigations. NIST Technical Series Publication NIST AI 100-2e2025, National Institute of Standards and Technology, Gaithersburg, MD.

Yu, G., Wang, Y., Wang, Z., Chen, L., Zheng, Y., and Liu, Y. (2026). Llms in industrial domains: A systematic review of adaptation techniques and applications from the product lifecycle perspective. *Advanced Engineering Informatics*, 74:104655.

Zheng, L., Chiang, W.-L., Sheng, Y., Zhuang, S., Wu, Z., Zhuang, Y., Lin, Z., Li, Z., Li, D., Xing, E. P., Zhang, H., Gonzalez, J. E., and Stoica, I. (2023). Judging LLM-as-a-Judge with MT-Bench and Chatbot Arena. In *Advances in Neural Information Processing Systems 36 (NeurIPS 2023) Datasets and Benchmarks Track*.

Zou, A., Wang, Z., Carlini, N., Nasr, M., Kolter, J. Z., and Fredrikson, M. (2023). Universal and transferable adversarial attacks on aligned language models. *arXiv preprint arXiv:2307.15043*.