Efetividade vs. Custo Computacional: Uma proposta para análise de ataques de sufixo adversarial GCG e nanoGCG

Carlos D. S Bunn[1], **Matheus R. S. Corrêa**[1], **Charles C. Miers**[1]

[1]Departamento de Ciência da Computação (DCC)
Universidade do Estado de Santa Catarina (UDESC)

{carlos.bunn,matheus.correa}@edu.udesc.br,

charles.miers@udesc.br

Resumo. *Com a adoção de Large Language Models (LLMs) em tarefas cotidianas, é necessário avaliar os riscos de segurança e conformidades de desempenho. Analisar estes sistemas é essencial para verificar seu alinhamento às diretrizes regulatórias. Este artigo apresenta uma proposta de análise comparativa entre os métodos de ataque adversarial baseados em sufixo Greedy Coordinate Gradients (GCGs) e nanoGCGs (nanoGCGs), mensurando custo computacional e efetividade adversarial no ambiente Python Risk Identification Tool (PyRIT) por meio de métricas como tempo de execução, consumo de memória e taxa de sucesso de ataque.*

1. Introdução

Os *Large Language Model*s (LLMs) são modelos treinados com grandes volumes de dados para compreender, gerar e manipular linguagem natural, o que lhes permite executar uma ampla gama de tarefas complexas [Zou et al. 2023]. Graças a essa versatilidade, sistemas de *chatbots* baseados nessas arquiteturas têm sido amplamente adotados em diversas aplicações modernas. Contudo, essa integração exige que tais sistemas operem sob princípios rigorosos de segurança e conformidade, seguindo diretrizes de entidades como o NIST [Vassilev et al. 2025] e a CSA [Alliance and Exchange 2025]. Apesar dessas salvaguardas, modelos de *Generative Artificial Intelligence* (GenAI) ainda podem ser induzidos a violar suas políticas de segurança por meio de ataques adversariais, como os baseados em geração de sufixos [Zou et al. 2023].

Dentre os diversos métodos, se destaca o *Greedy Coordinate Gradient*s (GCGs), que otimiza iterativamente o *prompt* para maximizar a probabilidade de geração de respostas proibidas. Essas abordagens, contudo, apresentam um elevado custo computacional por exigirem o cálculo de gradientes sobre o espaço latente de *tokens* para identificar o sufixo ideal. Como alternativa, variantes mais eficientes, como o *nanoGCG* (nanoGCG), foram propostas para reduzir o custo por iteração, mantendo os níveis de efetividade similares [**?**]. Diante deste cenário, este trabalho propõe uma investigação comparativa do custo computacional e da eficiência dos métodos GCG e nanoGCG, analisando métricas como tempo de execução, consumo de memória, Attack Success Rates (ASRs) e custo por ataque bem-sucedido. Para assegurar o rigor experimental e a rastreabilidade dos resultados, utiliza-se o *framework Python Risk Identification Tool*s (PyRITs) [Microsoft AI Red Team 2026] como ferramenta de avaliação, tendo como alvos os modelos GPT-4o e Llama 3.

O restante do artigo está organizado da seguinte forma: a Seção 2 apresenta a fundamentação teórica sobre GCG, nanoGCG e PyRIT; a Seção 3 detalha a metodologia de análise comparativa e os critérios utilizados; por fim, a Seção 4 traz as considerações finais e reflexões sobre os resultados obtidos.

2. Fundamentação

O método GCG calcula a escolha de novos sufixos a partir da representação dos *tokens* de entrada. O algoritmo opera de forma iterativa e gulosa, selecionando a substituição que produz a maior variação na probabilidade da saída desejada. Ao explorar a otimização coordenada, o GCG promove alterações de maior profundidade semântica [Zou et al. 2023], o que favorece a retenção da intenção adversarial.

Diferentemente da versão tradicional, o nanoGCG opera em janelas reduzidas de avaliação de candidatos. Essa abordagem implica menor profundidade semântica e pode reduzir a retenção da intenção adversarial ao selecionar apenas *tokens* previamente identificados como promissores [Winninger et al. 2025]. Contudo, essa limitação de contexto contribui para a redução do custo computacional e da sobrecarga na otimização. Em ambos os casos, o cálculo do gradiente em relação aos *prompts* orienta a atualização iterativa dos *tokens* até que o critério de parada seja atingido [Zou et al. 2023]. A distinção entre as abordagens reside, portanto, na extensão do contexto analisado, impactando a eficiência e a efetividade do ataque.

O *framework* PyRIT padroniza a execução de ataques adversariais, estruturando a submissão de *prompts* e a avaliação das respostas conforme critérios predefinidos [Microsoft AI Red Team 2026]. A Figura 1 ilustra o fluxo de execução integrado entre o PyRIT, o gerador de sufixos e o modelo alvo.

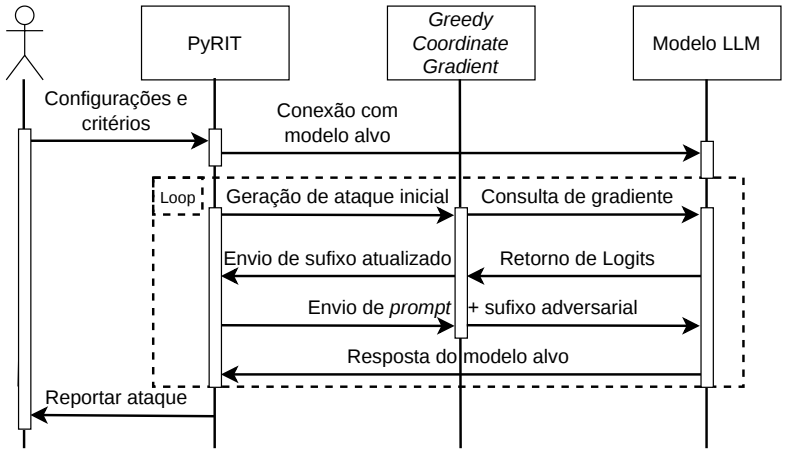

Figura 1. Sequência de sistema de análise do PyRIT.

O processo inicia-se com a definição de configurações pelo usuário, seguida pela conexão do PyRIT ao modelo para acionar o ataque. A ferramenta permite um processo iterativo de atualização do sufixo, utilizando informações como *logits* ou gradientes. O ciclo repete-se até que o critério de sucesso ou o limite de iterações seja alcançado, culminando na consolidação dos resultados.

3. Proposta e critérios

Este trabalho propõe uma avaliação comparativa entre os métodos GCG e nanoGCG, executados em um ambiente orquestrado pelo *framework* PyRIT. O objetivo central da análise é mensurar o desempenho computacional e a efetividade adversarial desses algoritmos quando são aplicados a LLM — especificamente o ChatGPT-4o e o Llama-3 (7B) — com a finalidade de quantificar os ganhos e perdas de cada uma das técnicas.

Para podermos assegurar a validade científica da comparação, ambos os métodos são submetidos a condições experimentais equivalentes, o que inclui o uso da mesma infraestrutura computacional, modelos-alvo idênticos, o mesmo conjunto de *prompts* base, além de limites de iterações e critérios de parada uniformes. Essa padronização visa garantir a integridade e a coerência da análise comparativa. O fluxo experimental planejado, que integra a execução dos ataques no ambiente PyRIT, está detalhado na Figura 2.

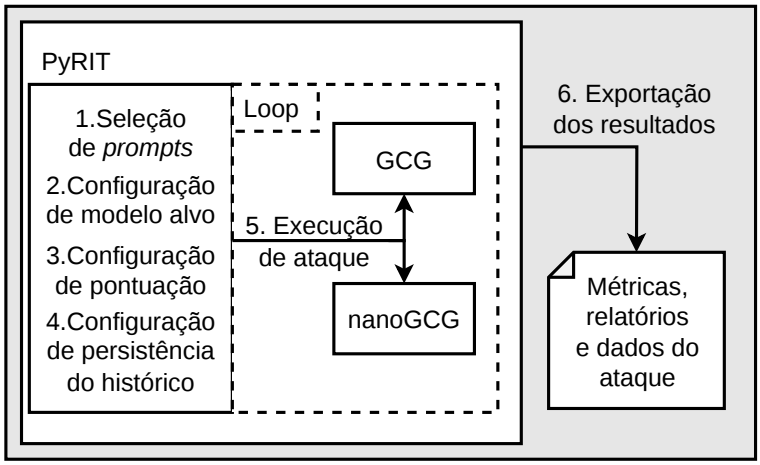

Figura 2. Estrutura de testes PyRIT para grupo *Greedy Coordinate Gradient*.

A investigação visa identificar disparidades no custo computacional e na eficácia de indução de respostas que violem políticas de segurança predefinidas. A avaliação fundamenta-se nas seguintes métricas quantitativas previamente definidas:

- **Tempo de execução por ataque**: tempo total, medido em segundos, compreendido entre a geração inicial do sufixo adversarial e o encerramento do processo iterativo.
- **Consumo de memória**: utilização média e máxima de memória principal durante a execução do algoritmo.
- **Número de iterações**: quantidade de ciclos executados até a obtenção de sucesso adversarial ou até o atingimento do limite máximo estabelecido.
- ***Attack Success Rate*ASR**: proporção entre o número de ataques que resultam em resposta classificada como violação de política e o número total de tentativas realizadas.
- **Custo por ataque bem-sucedido**: razão entre o tempo total de execução acumulado e o número de ataques que atingiram sucesso adversarial.

A comparação entre os cenários permite avaliar o impacto da extensão da janela de contexto e da profundidade das modificações adversariais no desempenho computaci-

onal e na efetividade do ataque. O critério de sucesso é definido pelo módulo avaliador do PyRIT com as informações do ataque. Essa análise torna viável compreensão das implicações práticas na adoção do GCG ou do nanoGCG em processos de avaliação de segurança de modelos de linguagem.

4. Considerações

O PyRIT fornece mecanismos que permitem analisar as diferenças entre os métodos GCG e nanoGCG. Dessa forma, torna-se possível aprofundar a avaliação do desempenho de ambos os algoritmos, identificando quais componentes influenciam a eficiência na geração de sufixos adversariais. A análise comparativa contribui para a compreensão do equilíbrio entre eficiência computacional e capacidade ordinária na indução da violações de políticas de segurança em modelos de linguagem de grande escala. Como continuidade, se prevê a execução dos experimentos descritos, que visam consolidar e fortalecer as metodologias de análise de segurança em sistemas baseados em LLM.

Agradecimentos: Os autores agradecem o apoio da LabP2D/UDESC e a FAPESC.

Referências

Alliance, C. S. and Exchange, O. A. (2025). *Agentic AI Red Teaming Guide*. Cloud Security Alliance. Disponível em: https://cloudsecurityalliance.org/artifacts/agentic-ai-red-teaming-guide. Acesso em: 14 out. 2025.

Microsoft AI Red Team (2026). Pyrit documentation: The python risk identification tool for generative ai. https://azure.github.io/PyRIT/. Acesso em: 20 fev. 2026.

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.

Winninger, T., Addad, B., and Kapusta, K. (2025). Using mechanistic interpretability to craft adversarial attacks against large language models. *arXiv preprint arXiv:2503.06269*.

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*.