# Avaliação da Herança de Propriedades Adversariais em Grandes Modelos de Linguagem com Restrição de Contexto e Destilação de Conhecimento Latente

**Gabriel de A. Campos**[1], **Martin Andreoni**[2], **Diogo M. F. Mattos**[1]

[1] LabGen/MídiaCom - PPGEET/TET/PGC/TCC
Universidade Federal Fluminense (UFF) - Brasil
[2]Technology Innovation Institute, 9639 Masdar City, Abu Dhabi - UAE

***Resumo.*** *Este trabalho propõe o arcabouço Prompt Inheritance, voltado à geração de ataques de jailbreaking em Grandes Modelos de Linguagem (LLMs) via destilação de conhecimento latente. A metodologia avalia a herança de propriedades adversariais comparando a manipulação do espaço de embeddings com a exposição a contextos restritos. Experimentos controlados sem ajuste de parâmetros indicam taxas de sucesso de até 61% com prompts concisos. Os resultados demonstram que a proposta supera abordagens da literatura em eficiência, exigindo significativamente menos recursos computacionais e viabilizando ataques escaláveis em hardware acessível.*

***Abstract.*** *This work proposes the Prompt Inheritance framework, aimed at generating jailbreaking attacks in Large Language Models (LLMs) through latent knowledge distillation. The methodology evaluates the inheritance of adversarial properties by comparing the manipulation of the embedding space with exposure to restricted contexts. Controlled experiments without parameter fine-tuning indicate success rates of up to 61% with concise prompts. The results demonstrate that the proposal outperforms literature approaches in efficiency, requiring significantly fewer computational resources and enabling scalable attacks on accessible hardware.*

## 1. Introdução

O ganho de maturidade da Inteligência Artificial Generativa deve-se muito aos Grandes Modelos de Linguagem (LLMs) e métodos de raciocínio como *Chain of Thought* [Wei et al. 2022]. Contudo, instruções projetadas para contornar salvaguardas, conhecidas como *jailbreak*, permanecem eficazes. Técnicas atuais de geração automática, como *AutoDAN-Turbo* [Liu et al. 2024], embora eficazes, exigem alto custo computacional e múltiplas iterações com agentes baseados em LLMs, demandando hardware de alto desempenho. Dessa maneira, a maioria das abordagens visam manipulações estruturais, como modificação de sintaxe ou inserção de sufixos [Zhu et al. 2024]. Todavia, a falta de modelagem semântica leva a processos de tentativa e erro que consomem recursos excessivos de processamento, CPU ou GPU.

Este trabalho foi financiado pelo Instituto de Ciência e Tecnologia Itaú (ICTi), CNPq, CAPES, FAPERJ, RNP e INCT ICONIOT. Ferramentas de Inteligência Artificial Generativa, incluindo ChatGPT, Grammarly e Perplexity, foram empregadas na revisão textual deste trabalho.

Este artigo apresenta o arcabouço `Prompt Inheritance`[1], que explora a herança semântica no espaço de *embeddings* para gerar *prompts* adversariais de baixo custo. Utilizando técnicas como Análise de Componentes Principais (PCA), Análise de Componentes Independentes (ICA), *Score-Weighted* PCA e *Gradient-Weighted Bag-of-Tokens*, o método extrai direções semânticas de ataques. Os resultados revelam taxas de sucesso de até 61% com redução drástica na infraestrutura necessária, demonstrando que a generalização intrínseca dos LLMs permite a automação de ataques sem a necessidade de processos iterativos exaustivos.

O restante deste artigo está organizado da seguinte forma. A Seção 2 revisa os trabalhos relacionados. A Seção 3 discute os ataques de *jailbreaking* e introduz os principais desafios de segurança em LLMs. A Seção 4 detalha o arcabouço *Prompt Inheritance*. A Seção 5 discute os resultados experimentais e a eficácia da herança semântica e a Seção 6 conclui este trabalho.

## 2. Trabalhos Relacionados

Os métodos de geração de *prompts* adversariais evoluíram de abordagens manuais para arcabouços automatizados. O GCG (*Greedy Coordinate Gradient*) [Zou et al. 2023b] introduz uma estratégia de busca baseada em gradientes para identificar sufixos que maximizam a probabilidade de aceitação de comandos maliciosos por LLMs. Apesar de sua eficácia, o método apresenta como limitação a geração frequente de textos semanticamente ininteligíveis. Em resposta a essa limitação, arcabouços como o AutoDAN-Turbo [Liu et al. 2024] empregam agentes baseados em LLMs para o refinamento iterativo de *prompts*, alcançando elevada fluência textual, porém com custos computacionais significativamente mais elevados, decorrentes do processo iterativo e da dependência de infraestrutura de alto desempenho. Paralelamente, a análise do comportamento interno dos modelos tem emergido como uma linha de pesquisa relevante. O conceito de Engenharia de Representação (*Representation Engineering* – RepE) [Zou et al. 2023a] propõe a manipulação direta dos estados internos dos modelos (*embeddings*) como mecanismo de controle comportamental, em contraste com abordagens que atuam exclusivamente sobre o texto de entrada. Técnicas de decomposição, como PCA e ICA, têm sido exploradas para analisar e navegar no espaço de *embeddings*, permitindo a identificação de direções latentes associadas a atributos semânticos específicos e ao comportamento do modelo.

Em avanços recentes, a segurança de LLMs tem sido abordada por mecanismos de defesa como o ARMOR [Zhao et al. 2025], que utiliza verificação em tempo de execução, e *benchmarks* como o JailbreakRadar [Chu et al. 2025], que estabelece uma taxonomia para avaliar robustez de modelos frente a ataques adaptativos. No campo ofensivo, técnicas como CAVGAN [Li et al. 2025] manipulam vetores de ativação latente via Redes Adversárias Generativas (GANs), enquanto o método IRIS [Huang et al. 2025] foca na supressão de mecanismos de recusa para aumentar a transferência de ataques. O *Prompt Inheritance* diferencia-se ao focar na eficiência operacional: em vez de treinar redes complexas ou realizar otimizações iterativas, ele reutiliza o conhecimento latente de ataques preexistentes, posicionando-se como uma alternativa de baixo custo computacional entre a otimização textual pura e a manipulação direta de estados internos.

[1]Disponível em: `https://github.com/gdac7/prompt-inheritance`

## 3. Ataques de *Jailbreaking* e Desafios de Segurança em LLMs

Os Grandes Modelos de Linguagem (*Large Language Models* - LLMs) consolidaram-se como componentes centrais de sistemas de *software* contemporâneos ao estabelecer a linguagem natural como principal interface para a realização de tarefas complexas. Essa centralidade, contudo, introduz desafios de segurança estruturais, associados sobretudo à fragilidade na separação entre instruções do sistema e entradas fornecidas pelo usuário. Embora modelos recentes sejam submetidos a extensivos processos de alinhamento, como o Aprendizado por Reforço com *Feedback* Humano (*Reinforcement Learning from Human Feedback* - RLHF), tais mecanismos operam de forma probabilística e não oferecem garantias determinísticas contra entradas adversariais cuidadosamente elaboradas.

A superfície de ataque dos LLMs decorre diretamente de sua capacidade de processamento semântico. A mesma flexibilidade que permite interpretar variações linguísticas também possibilita a ocultação de intenções maliciosas em contextos aparentemente benignos. À medida que esses modelos passam a integrar agentes autônomos com permissões de leitura, escrita e execução em sistemas críticos, ataques baseados em *prompting* deixam de ser uma preocupação restrita à geração de conteúdo e tornam-se vetores de vulnerabilidades sistêmicas. Nesse cenário, o *jailbreaking* caracteriza-se como uma classe de ataques adversariais, predominantemente de natureza *black-box*, na qual entradas estrategicamente construídas subvertem mecanismos de alinhamento e salvaguardas de segurança, explorando a própria capacidade do modelo de compreender e gerar linguagem natural para induzi-lo a violar restrições internas e diretrizes éticas.

Padrões recorrentes desses ataques incluem a manipulação de *personas*, o encadeamento de instruções e a exploração de ambiguidades semânticas no espaço de *embeddings*. O método *Do Anything Now* (DAN) ilustra esse fenômeno ao explorar a dissonância entre o seguimento de instruções e o alinhamento de segurança, induzindo o modelo a priorizar a obediência a comandos do usuário por meio de narrativas fictícias e mecanismos persuasivos [Zhu et al. 2024]. Abordagens mais recentes, como AutoDAN, demonstram que é possível identificar *prompts* adversariais de forma sistemática: enquanto variantes *white-box* utilizam otimização por gradiente para maximizar a aceitação de requisições maliciosas [Zhu et al. 2024], versões majoritariamente *black-box* recorrem a arquiteturas multiagente e refinamento iterativo de instruções [Liu et al. 2024]. A subversão de salvaguardas pode transformar LLMs em vetores para exfiltração de dados, manipulação de APIs e comprometimento de infraestruturas, especialmente quando integrados a ferramentas externas e fluxos automatizados. As defesas atualmente empregadas, baseadas em filtragem de *prompts* e alinhamento via RLHF, apresentam limitações estruturais e baixa capacidade de generalização frente a ataques adaptativos, funcionando frequentemente como barreiras superficiais contornáveis por variações fora da distribuição de treinamento. Diante desse contexto, o estudo do *jailbreaking* é essencial não apenas como prática de *red teaming*, mas como instrumento para revelar limites fundamentais das arquiteturas de atenção e orientar a transição de estratégias reativas para o desenvolvimento de modelos inerentemente mais robustos e confiáveis.

## 4. Arcabouço *Prompt Inheritance*

O arcabouço *Prompt Inheritance* opera através de um *pipeline* modular composto por quatro etapas principais, ilustatradas na Figura 1, que transformam uma requisição maliciosa em um *prompt* de ataque funcional: (i) Busca por Similaridade, (ii) Identificação da


**Figura 1. Arcabouço *Prompt Inheritance* para geração de *prompts* adversariais. O processo parte de um pedido malicioso e percorre as etapas de Busca por Similaridade, Identificação da Essência Adversarial, Construção do Bag-of-Tokens (BoT) e Sanitização, resultando em um prompt final fluente que preserva a intenção adversarial por meio da herança semântica.**

Essência Adversarial (IEA), (iii) Construção do Bag-of-Tokens (BoT) e (iv) Sanitização. As etapas e componentes do arcabouço são discutidos a seguir.

A **Busca por Similaridade** reduz a base de *prompts* para os mais similares ao contexto do pedido malicioso. A etapa de **Identificação da Essência Adversarial (IEA)** tem como finalidade extrair os atributos semânticos fundamentais que caracterizam o comportamento adversarial presente no conjunto reduzido de *prompts*. Diferentemente de abordagens puramente textuais, essa etapa opera diretamente sobre as representações vetoriais, buscando capturar padrões latentes associados à eficácia dos ataques. Para isso, o arcabouço *Prompt Inheritance* propõe quatro abordagens distintas de decomposição, manipulação e análise, cada uma explorando diferentes propriedades do espaço de *embeddings*.

As abordagens de **Decomposição por PCA e ICA** constroem um vetor latente a partir da busca por similaridade. Inicialmente, calcula-se o centroide para centralizar as representações, dado por:

$$\bar{\mathbf{c}} = \frac{\mathbf{1}}{\mathbf{n}} \sum_{\mathbf{i=1}}^{\mathbf{n}} \mathbf{e_i}, \tag{1}$$

onde $\bar{c}$ é o vetor médio. Após a centralização, extrai-se o componente $v_1$. No PCA, $v_1$ é o componente de maior variância; no ICA, a primeira fonte estatisticamente independente. Após a centralização das representações vetoriais, procede-se à extração do componente $v_1$, que concentra a principal direção latente do conjunto de *embeddings*.

No caso da decomposição por PCA, $v_1$ corresponde ao primeiro componente principal, responsável por capturar a direção de maior variância dos dados. Já na decomposição por ICA, $v_1$ representa a primeira fonte estatisticamente independente identificada, buscando modelar um padrão semântico maximalmente não gaussiano e independente das demais fontes extraídas. Essa distinção reflete diferentes interpretações da estrutura latente dos *prompts*, permitindo explorar tanto variações dominantes quanto independências semânticas.

A partir do componente $v_1$, define-se o deslocamento do vetor latente por:

$$\bar{\mathbf{c}}_{\mathbf{new}} = \mathbf{centroide} + \alpha \mathbf{v_1}, \tag{2}$$

em que o hiperparâmetro $\alpha$ controla a magnitude e a orientação da geração do novo ve-

tor $c_{new}$ em relação ao centroide. O sinal de $\alpha$ define a orientação do deslocamento (na mesma direção de $v_1$ ou oposta), permitindo explorar regiões semanticamente contrastantes, enquanto $\alpha = 0$ mantém o vetor no centroide original, servindo como ponto de referência neutro.

O hiperparâmetro $\alpha$ corresponde ao desvio padrão dos dados projetados em $v_1$, garantindo que o deslocamento no espaço latente respeite a escala da distribuição original e preserve a coerência semântica. No PCA, $\alpha$ provém da variância explicada; no ICA, o dado é explicitamente projetado na direção independente para o cálculo. Essa padronização assegura que o deslocamento seja estatisticamente comparável entre os métodos, servindo de base para as extensões ponderadas apresentadas a seguir.

O método **Score-Weighted PCA (SWPCA)** introduz uma ponderação baseada no sucesso histórico dos ataques para guiar a herança semântica. Enquanto o PCA convencional atribui relevância igual a todos os *prompts*, o SWPCA prioriza vetores eficazes para reduzir a influência de ruído e instruções pouco produtivas [Lyu and Yuan 2025]. Essa estratégia direciona a análise para regiões do espaço latente correlacionadas a taxas de sucesso elevadas.

O processo inicia-se com a aplicação de um limiar de pontuação $\theta$, que filtra o conjunto de *embeddings* de forma a reter apenas amostras associadas a alto desempenho. Para cada vetor filtrado, calcula-se um peso $w_i$ proporcional à sua pontuação, normalizado, dado por

$$\mathbf{w_i} = \frac{\mathbf{s_i} - \mathbf{s_{min}}}{\mathbf{s_{max}} - \mathbf{s_{min}}}, \tag{3}$$

em que $s_i$ representa a pontuação do $i$-ésimo *prompt*, enquanto $s_{min}$ e $s_{max}$ denotam, respectivamente, as pontuações mínima e máxima do conjunto selecionado. Essa normalização assegura que os pesos reflitam diferenças relativas de desempenho, mantendo comparabilidade entre as amostras.

A partir desses pesos, define-se um centroide ponderado $\bar{c}_{weighted}$, dado por

$$\bar{\mathbf{c}}_{\mathbf{weighted}} = \frac{\sum \mathbf{i} = \mathbf{1^n} \mathbf{w_i} \mathbf{e_i}}{\sum_{\mathbf{i=1}}^{\mathbf{n}} \mathbf{w_i}}, \tag{4}$$

que substitui a média aritmética simples empregada nos métodos não ponderados (Equação 1). Ao incorporar explicitamente a eficácia histórica dos ataques no cálculo do centroide, essa formulação desloca a referência semântica para regiões do espaço latente associadas a *prompts* mais impactantes, estabelecendo uma base mais informativa para a decomposição subsequente.

Com o centroide ponderado definido, a análise de componentes principais é realizada sobre uma matriz de covariância igualmente ponderada, seguindo a abordagem de decomposição espectral para dados ponderados [Delchambre 2014]. Essa técnica permite que o componente $v_1$ capture a direção de variância mais relevante sob a perspectiva da pontuação dos ataques, em vez de refletir apenas a dispersão global dos dados. Como consequência, o deslocamento $\alpha$ aplicado sobre $\bar{c}_{weighted}$ resulta em um vetor $\bar{c}_{new}$:

$$\bar{\mathbf{c}}_{\mathbf{new}} = \bar{\mathbf{c}}_{\mathbf{weighted}} + \alpha \mathbf{v_1}, \tag{5}$$

que herda não apenas a similaridade semântica com os *prompts* originais, mas também atributos estruturais fortemente correlacionados a ataques previamente avaliados com altas pontuações.

Diferentemente das abordagens baseadas em decomposição estatística, o método **Gradient-Weighted Bag-of-Tokens (GWBoT)** emprega retropropagação para quantificar a influência direta de *tokens* individuais na probabilidade de aceitação da requisição pelo modelo alvo. Essa técnica inspira-se em ataques baseados em gradiente, como GCG [Zou et al. 2023b] e AutoDAN [Zhu et al. 2024], porém é aplicada especificamente ao conjunto de *prompts* recuperados na etapa de Busca por Similaridade, com o objetivo de identificar termos com elevada sensibilidade adversarial. Ao focar na contribuição individual dos *tokens*, o GWBoT explora uma perspectiva complementar às abordagens latentes, fundamentada na resposta diferenciada do modelo a cada elemento textual.

O princípio central do procedimento adotado é formalizado por:

$$\mathcal{L}(\mathbf{e_{dummy}}) = -\sum_{\mathbf{j=1}}^{|\mathbf{T}|} \log \mathbf{P}(\mathbf{t_j} \mid \mathbf{x}, \mathbf{e_{dummy}}, \mathbf{t_1}, \ldots, \mathbf{t_{j-1}}; \theta), \tag{6}$$

em que define-se inicialmente uma resposta alvo afirmativa $T$, tipicamente uma expressão como "Sure, here is", frequentemente observada quando modelos de linguagem atendem a pedidos do usuário. O objetivo consiste em minimizar a função de perda de entropia cruzada $\mathcal{L}$ entre a saída do modelo e o alvo prefixado. Essa perda é calculada a partir da inserção de um *token* variável $e_{dummy}$ ao final da requisição maliciosa $x$, permitindo avaliar a sensibilidade do modelo a perturbações locais no espaço de *embeddings*. O vetor $\theta$ representa os parâmetros do modelo, de modo que a aplicação dessa técnica pressupõe acesso às informações internas do modelo de linguagem, caracterizando um cenário de *White-box Large Language Models*.

A partir do cálculo da perda, extrai-se o gradiente em relação ao *embedding* de entrada associado ao *token* variável, definido como $g = \nabla_{e_{dummy}} \mathcal{L}$. Esse vetor gradiente indica a direção no espaço de *embeddings* que mais reduz a resistência do modelo à requisição maliciosa. Em seguida, o arcabouço calcula o produto escalar entre o gradiente negativo e os *embeddings* de todos os *tokens* presentes nos *prompts* selecionados na etapa de Busca de Similaridade, denotados por $E_{cand}$. Para cada *token* candidato $e_i \in E_{cand}$, atribui-se uma pontuação de influência $\phi_i$, conforme:

$$\phi_{\mathbf{i}} = \langle \mathbf{e_i}, -\mathbf{g} \rangle. \tag{7}$$

*Tokens* com valores mais elevados de $\phi_i$ são interpretados como aqueles com maior capacidade de induzir o modelo ao estado de aceitação desejado. Diferentemente das abordagens baseadas em decomposição, esse método não gera um novo vetor latente, mas produz uma ordenação de *tokens* reais pré-existentes segundo sua relevância adversarial, os quais são posteriormente agregados em um *Bag-of-Tokens* para sanitização.

A etapa de **Construção do Bag-of-Tokens (BoT)** tem como finalidade definir o conjunto de *tokens* que será utilizado como contexto na criação do *prompt* adversarial. Para os métodos baseados em espaço latente, emprega-se a técnica de Decodificação Ortogonal Iterativa, que seleciona iterativamente os *tokens* cujos *embeddings* apresentam maior similaridade de cosseno com o vetor gerado. A cada iteração, aplica-se uma penalidade de projeção ortogonal para remover componentes semânticas já exploradas, assegurando diversidade no conjunto selecionado. O vetor alvo $v$ é atualizado após a seleção de cada token $t$ por:

$$\mathbf{v_{next}} = \mathbf{v_{current}} - \left( \frac{\mathbf{v_{current}} \cdot \mathbf{e_t}}{|\mathbf{e_t}|^2} \right) \mathbf{e_t} \cdot \gamma, \tag{8}$$

em que $e_t$ representa o *embedding* do *token* escolhido e $\gamma$ é um fator de decaimento que controla a intensidade da remoção semântica.

No caso específico do método GWBoT, a construção do BoT ocorre de forma direta, selecionando-se os $k$ *tokens* com maior pontuação de influência $\phi$. O resultado final dessa etapa é o *Bag-of-Tokens* $B$, de tamanho arbitrário $k$, composto por uma lista de termos que preserva o objetivo e o potencial adversarial do ataque. Esse conjunto serve como elo entre a extração de características adversariais e a geração textual final.

Por fim, na etapa de **Sanitização**, o conhecimento adversarial extraído do conjunto base de *prompts* é concentrado no BoT aprendido na etapa anterior para a geração de um texto fluido, coerente e semanticamente consistente. Nessa fase, um modelo de linguagem é empregado para transformar o conjunto não estruturado de *tokens* em um *prompt* natural e bem formado. Neste trabalho, utilizou-se o modelo `Llama-3.1-8B-Instruct`, da Meta, encerrando o *pipeline* de geração adversarial ao converter informação latente e tokenizada em linguagem natural explícita.

## 5. Resultados e Discussão

Para avaliar a hipótese de que as abordagens propostas são capazes de herdar características adversariais, analisa-se a distribuição espacial dos vetores gerados no espaço semântico. A Figura 2 apresenta a projeção bidimensional dos *embeddings* dos *Bag-of-Tokens* produzidos, contrastados com os $M$ *prompts* utilizados como base de destilação. Na visualização, os pontos em escala de cinza representam os ataques originais, em que tons mais escuros indicam maiores pontuações de eficácia. Observa-se um alinhamento consistente entre os vetores gerados e as regiões de maior densidade da base, o que sugere que as abordagens propostas navegam de forma direcionada pelo espaço latente. Esse comportamento estabelece o fundamento para a análise comparativa entre os diferentes métodos de herança semântica, discutida a seguir.

A Figura 2 confirma que PCA, ICA e SWPCA mantêm a concentração no quadrante semântico da base, com o SWPCA deslocando-se sistematicamente para regiões de maior eficácia. Esses resultados indicam a capacidade das abordagens de decomposição em navegar de forma direcionada pelo espaço latente para extrair a essência adversarial.

Em contraste, o método GWBoT apresenta uma dispersão mais acentuada quando comparado às abordagens de decomposição estatística. Diferentemente do PCA e do ICA, que operam sobre centroides semânticos e direções latentes contínuas, o GWBoT

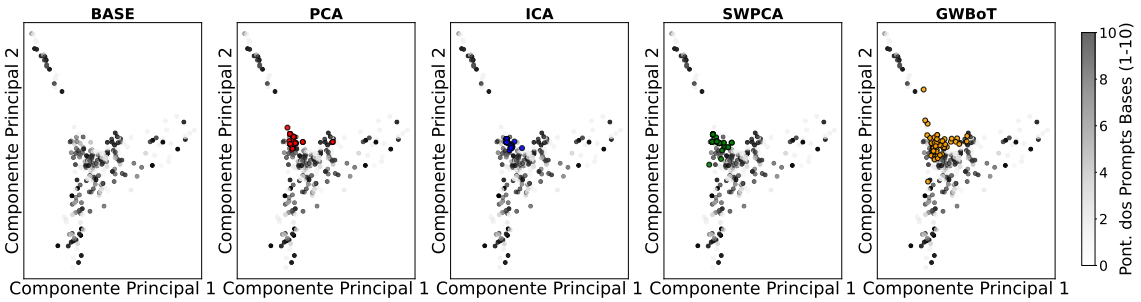


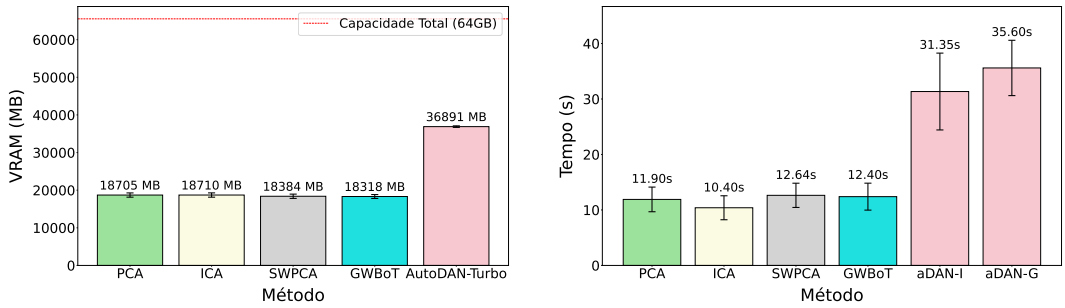

**Figura 2. Projeção bidimensional do espaço semântico dos *embeddings*, comparando os *prompts* base e os *Bag-of-Tokens* gerados pelas abordagens PCA, ICA, SWPCA e GWBoT. Os pontos em escala de cinza representam os ataques originais, com tonalidade proporcional à pontuação de eficácia, enquanto os pontos coloridos indicam os vetores gerados, evidenciando seu alinhamento com regiões de alta densidade semântica.**

seleciona *tokens* discretos com base na maximização do gradiente negativo em relação à resposta alvo. Esse mecanismo resulta na extração de termos que, embora possam ser semanticamente distantes entre si no espaço vetorial, exercem elevada influência individual na redução da função de perda. Ainda que essa abordagem não produza um novo vetor latente $\bar{c}_{new}$, observa-se uma tendência consistente de concentração na mesma região do espaço semântico identificada pelos métodos anteriores, reforçando sua capacidade de capturar atributos adversariais.

A viabilidade de arcabouço de *red teaming* e geração de ataques adversariais em ambientes reais é frequentemente limitada pela exigência de *hardware* de alto desempenho. Avalia-se, então, a eficiência do arcabouço *Prompt Inheritance* sob a perspectiva do consumo de memória de vídeo (VRAM), comparando-o diretamente com o método de referência AutoDAN-Turbo. O método base emprega quatro LLMs que atuam de forma dinâmica na síntese de estratégias responsáveis por guiar a formação dos *prompts* adversariais, o que resulta em uma demanda computacional significativamente elevada. Essa característica motiva a análise comparativa proposta, estabelecendo o contexto para a avaliação de eficiência de recursos apresentada a seguir.

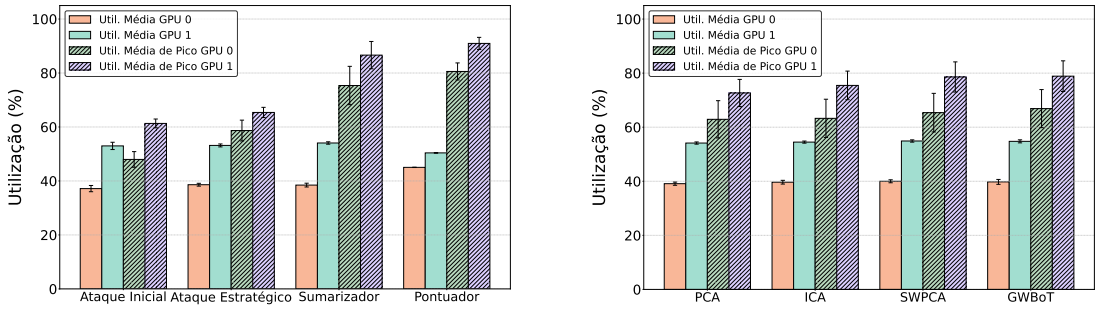
Para viabilizar a execução do AutoDAN-Turbo, os módulos Atacante, Sumarizador, Alvo e Pontuador foram distribuídos em dois nós equipados com 2x GPUs RTX 5060 Ti 16gb (primeiro nó) e 2x 4060 Ti 16gb (segundo nó). Essa infraestrutura multi-GPU foi necessária apenas para o método base, evidenciando sua alta demanda em comparação ao *Prompt Inheritance*. Conforme a Figura 3(a), o AutoDAN-Turbo demandou cerca de 37 GB de VRAM, exigindo *hardware* robusto. Em contraste, as variantes do *Prompt Inheritance* mantiveram consumo estável entre 18 GB e 19 GB. Essa redução de aproximadamente 50% evidencia que a navegação direcionada no espaço de *embeddings* viabiliza a geração de *prompts* adversariais em *hardware* significativamente mais acessível, estabelecendo um ganho prático relevante em termos de viabilidade operacional.

A Figura 3(b) compara o tempo médio de geração. O AutoDAN-Turbo possui fluxos distintos: ataque inicial (aDAN-I) e guiado por estratégias prévias (aDAN-G), introduzindo estocasticidade. O *Prompt Inheritance*, por sua vez, apresenta menor comple-




(a) Uso médio de VRAM (MB). (b) Tempo de geração de um *prompt* Adversarial.

**Figura 3. Comparação do uso médio de VRAM (a) e do tempo médio de geração de *prompts* adversariais (b) entre o *Prompt Inheritance* e o método AutoDAN-Turbo. As propostas implicam a redução significativa no consumo de memória e no tempo de execução em relação ao modelo base.**




(a) Utilização média de computação das GPUs dos módulos do AutoDAN-Turbo.

(b) Utilização média de computação das GPUs das abordagens propostas.

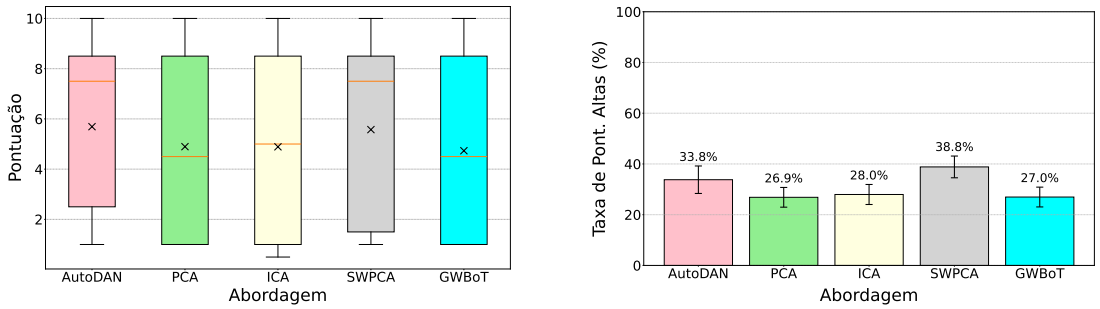
**Figura 4. Comparação da utilização média de computação das GPUs entre o AutoDAN-Turbo e as abordagens propostas.**

xidade estrutural e maior previsibilidade, alcançando tempos de geração substancialmente reduzidos em comparação a ambos os cenários do método referência.

As Figuras 4(a) e 4(b) comparam a carga de trabalho imposta ao *hardware*. As barras sólidas indicam a utilização média e as hachuras os valores médios de pico. O *Prompt Inheritance* manteve a utilização em cerca de 60%, enquanto o AutoDAN-Turbo excedeu 80%, especialmente na etapa de pontuação. Essa eficiência traduz-se em menor consumo energético e maior viabilidade em ambientes com recursos limitados.

Para avaliar a eficácia do arcabouço *Prompt Inheritance*, as respostas geradas pelo modelo alvo foram submetidas a uma métrica de pontuação automatizada destinada a mensurar o potencial adversarial dos *prompts* produzidos. Em conformidade com a metodologia e os critérios de avaliação estabelecidos no trabalho original do AutoDAN-Turbo [Liu et al. 2024], utilizou-se o modelo google/gemma-1.1-7b-it como juiz para atribuir notas em uma escala de 1 a 10. Nessa escala, a pontuação mínima corresponde a uma recusa completa da requisição por parte do modelo alvo, enquanto a pontuação máxima indica a aceitação integral do pedido malicioso original. A Figura 5(a) apresenta a distribuição das pontuações obtidas pelas diferentes abordagens avaliadas, em comparação com o modelo base AutoDAN-Turbo.

Observa-se uma elevada dispersão nas pontuações obtidas por todas as abordagens avaliadas, com resultados que abrangem praticamente toda a amplitude da escala. Essa




(a) Pontuações dos *prompts* gerados. (b) Taxa de *prompts* gerados com sucesso.

**Figura 5. Comparação entre a distribuição das pontuações e a taxa de sucesso dos *prompts* gerados.**

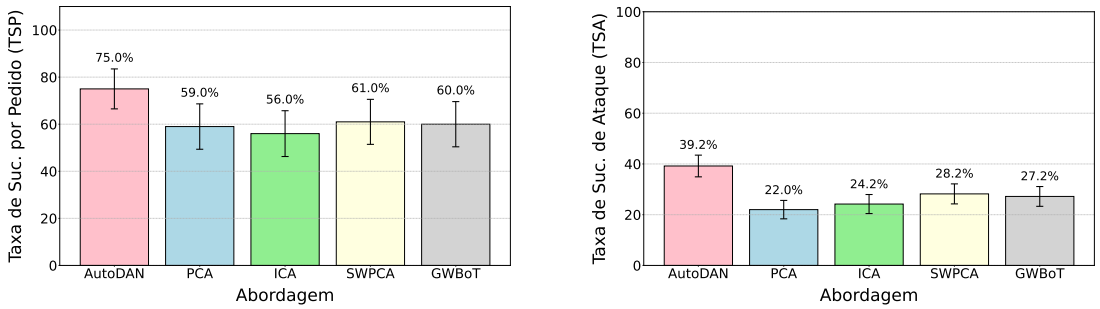
variabilidade acentuada indica que a eficácia de um ataque adversarial não é uniforme, mesmo quando fundamentada em atributos semânticos herdados de *prompts* semelhantes. A presença de pontuações extremas e distantes entre si reforça a natureza estocástica dos LLMs, evidenciando que o processo de geração de *prompts* adversariais se comporta, em grande parte, como uma operação de tentativa e erro. Nesse contexto, pequenas variações na estrutura textual ou na composição do *Bag-of-Tokens* podem conduzir a resultados drasticamente distintos.

Entre as abordagens analisadas, o método SWPCA apresentou, ao lado do AutoDAN, a maior mediana de pontuação (7.5), sugerindo que a captura da direção de maior variância semântica, ponderada pela pontuação, constitui uma estratégia promissora para a generalização de *tokens* adversariais. As técnicas PCA, ICA e GWBoT exibem medianas inferiores, refletindo o impacto da exploração semântica em um contexto não ponderado.

Essa distribuição de resultados sugere que a principal barreira para o sucesso consistente dos ataques ainda reside na sensibilidade contextual do modelo alvo, exigindo que o arcabouço adversarial lide com uma margem considerável de incerteza na eficácia individual dos *prompts* gerados. No âmbito do algoritmo AutoDAN-Turbo, instruções maliciosas que atingem uma pontuação igual ou superior a 8.5 são classificadas como casos de sucesso, por apresentarem maior probabilidade de caracterizar um *jailbreak*. A Figura 5(b) apresenta a proporção de *prompts* que superaram esse limiar de pontuação, servindo como base para a análise quantitativa de sucesso discutida a seguir.

Para avaliar *jailbreaks* efetivamente executados, adota-se a métrica Taxa de Sucesso de Ataque (TSA) [Mazeika et al. 2024]. Essa métrica é calculada por meio de um modelo Llama-2-13B *fine-tuned*, responsável por classificar a resposta do modelo alvo como recusa ou concordância em relação ao pedido malicioso. O TSA quantifica, portanto, a proporção de *prompts* que resultam efetivamente em *jailbreak*. Complementarmente, avalia-se a Taxa de Sucesso por Pedido (TSP), que, diferentemente da análise individual do TSA, mensura a proporção de pedidos maliciosos para os quais ao menos um *prompt* gerado conseguiu executar o *jailbreak*. No experimento conduzido, as técnicas propostas foram avaliadas em 100 requisições maliciosas provenientes do conjunto de dados HarmBench[2] , sendo gerados cinco *prompts* adversariais para cada técnica. Essa configuração experimental estabelece a base para a análise comparativa de sucesso

[2]Disponível em `https://www.harmbench.org/explore`.




(a) Taxa de Sucesso por Pedido (TSP). (b) Taxa de Sucesso de Ataque (TSA).

**Figura 6. Comparação entre a Taxa de Sucesso por Pedido (TSP) e a Taxa de Sucesso de Ataque (TSA).**

apresentada na sequência.

A Figura 6(a) evidencia que as abordagens apresentaram desempenhos similares. Esse equilíbrio sugere que os métodos orientaram a construção de *prompts* adversariais a partir de conceitos semânticos semelhantes, apesar das diferenças nos critérios de decomposição adotados. Por sua vez, o método SWPCA demonstrou que a ponderação baseada no desempenho histórico dos *prompts* é mais eficaz para a reprodução de características adversariais relevantes, alcançando a maior TSP entre as abordagens propostas, com 61%.

No que se refere aos resultados da Taxa de Sucesso de Ataque (TSA), apresentados na Figura 6(b), observa-se a proporção de *prompts* que efetivamente resultaram em *jailbreak*. Em ambas as avaliações, o método de referência AutoDAN-Turbo mantém desempenho superior, o que contrasta com os resultados apresentados na Figura 5(b), em que o SWPCA havia exibido pontuações médias mais elevadas. Essa discrepância evidencia limitações do avaliador baseado exclusivamente em pontuação e ressalta a necessidade de mecanismos mais eficazes para a identificação de estruturas adversariais genuínas, reforçando a importância do uso de métricas complementares na avaliação de *jailbreaks*.

## 6. Conclusão

Esse trabalho propôs e avaliou o arcabouço *Prompt Inheritance*, uma abordagem para a geração de *prompts* adversariais baseada na destilação de conhecimento prévio no espaço latente. A proposta foi motivada pela necessidade de alternativas mais eficientes aos processos iterativos de *red teaming*, os quais frequentemente impõem custos computacionais elevados e limitam sua adoção em ambientes práticos. Os resultados experimentais demonstram que o arcabouço é capaz de reduzir o consumo de memória de vídeo (VRAM) em aproximadamente 50%, evidenciando ganhos significativos de eficiência e viabilidade operacional em comparação a métodos baseados em múltiplos modelos iterativos. A técnica *Score-Weighted PCA* (SWPCA) destacou-se ao alcançar uma Taxa de Sucesso por Pedido (TSP) de 61%, corroborando a hipótese de que a ponderação por desempenho histórico favorece a captura da essência adversarial mais relevante. Apesar desses avanços, a análise da distribuição das pontuações revelou uma variabilidade acentuada na eficácia dos *prompts* gerados, reforçando que o processo de *jailbreak* permanece fortemente influenciado pela natureza probabilística dos LLMs. Essa característica impõe limites à previsibilidade dos ataques e evidencia a necessidade de mecanismos adicio-

nais para reduzir a incerteza associada à geração adversarial. Como trabalhos futuros, pretende-se investigar a transição da extração de *tokens* isolados para a geração de frases completas e semanticamente estruturadas, com o objetivo de aumentar simultaneamente a fluência textual e a eficácia dos ataques.

## Referências

Chu, J., Liu, Y., Yang, Z., Shen, X., Backes, M., and Zhang, Y. (2025). Jailbreakradar: Comprehensive assessment of jailbreak attacks against llms.

Delchambre, L. (2014). Weighted principal component analysis: a weighted covariance eigendecomposition approach. *Monthly Notices of the Royal Astronomical Society*.

Huang, D., Shah, A., Araujo, A., Wagner, D., and Sitawarin, C. (2025). Stronger universal and transferable attacks by suppressing refusals. In Chiruzzo, L., Ritter, A., and Wang, L., editors, *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 5850–5876, Albuquerque, New Mexico. Association for Computational Linguistics.

Li, X., Ning, Y., Bao, Z., Xu, M., Chen, J., and Qian, T. (2025). CAVGAN: Unifying jailbreak and defense of LLMs via generative adversarial attacks on their internal representations. In Che, W., Nabende, J., Shutova, E., and Pilehvar, M. T., editors, *Findings of the Association for Computational Linguistics: ACL 2025*, pages 6664–6678, Vienna, Austria. Association for Computational Linguistics.

Liu, X., Li, P., Suh, E., Vorobeychik, Y., Mao, Z., Jha, S., McDaniel, P., Sun, H., Li, B., and Xiao, C. (2024). AutoDAN-Turbo: A lightweight and scalable framework for automated jailbreaking. *arXiv preprint arXiv:2402.16439*.

Lyu, Z. and Yuan, M. (2025). Large-dimensional factor analysis with weighted pca. *arXiv preprint*.

Mazeika, M. et al. (2024). Harmbench: A standardized evaluation framework for automated red teaming and robust refusal.

Wei, J., Wang, X., Schuurmans, D., Bosma, M., Ichter, B., Xia, F., Chi, E. H., Le, Q. V., and Zhou, D. (2022). Chain-of-thought prompting elicits reasoning in large language models. In *Advances in Neural Information Processing Systems*, volume 35, pages 24824–24837. Curran Associates, Inc.

Zhao, Z., Ma, Y., Jha, S., Pavone, M., McDaniel, P., and Xiao, C. (2025). Armor: Aligning secure and safe large language models via meticulous reasoning.

Zhu, S., Zhang, R., et al. (2024). AutoDAN: Interpretable gradient-based adversarial attacks on large language models. *arXiv preprint*.

Zou, A., Phan, L., Chen, S., Campbell, J., Guo, P., Rens, R., Pan, A., Yin, X., Mazeika, M., Dombrowski, A.-K., et al. (2023a). Representation engineering: A top-down approach to AI transparency. *arXiv preprint*.

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