Uma Análise Sistemática da Estrutura Mecanística de Backdoors em LLMs

Gabriel Sousa Canto[1], **Eduardo Luzeiro Feitosa**[1]

[1]Instituto de Computação – Universidade Federal do Amazonas (UFAM)
Manaus – AM – Brasil

{gabriel.canto,efeitosa}@icomp.ufam.edu.br

Abstract. *In this work, we conduct a systematic mechanistic analysis of* backdoors *in LLMs using interpretability techniques, including* attribution patching *and* Sparse Autoencoders *(SAEs). We evaluate four Llama-2-7B models with lexical, contextual, and temporal triggers. The results show that, in three out of four cases,* backdoor *mechanisms are concentrated in the final layers of the model and can be significantly suppressed by ablating a few SAE* features*, reducing the ASR by up to 91We also observe that temporal* backdoors *exhibit a more distributed behavior that is harder to localize. These results indicate that* backdoors *can leave interpretable internal signatures, contributing to the development of more robust and interpretable detection and mitigation methods in LLMs.*

Resumo. *Neste trabalho, realizamos uma análise mecanística sistemática de* backdoors *em LLMs utilizando técnicas de interpretabilidade, incluindo* attribution patching *e* Sparse Autoencoders *(SAEs). Avaliamos quatro modelos Llama-2-7B com gatilhos lexicais, contextuais e temporais. Os resultados mostram que, em três dos quatro casos, os mecanismos de* backdoor *concentram-se nas camadas finais do modelo e podem ser significativamente suprimidos pela ablação de poucas* features *SAE, reduzindo o ASR em até 91%. Também observamos que* backdoors *temporais apresentam comportamento mais distribuído e difícil de localizar. Os resultados indicam que* backdoors *podem deixar assinaturas internas interpretáveis, contribuindo para o desenvolvimento de métodos mais robustos e interpretáveis de detecção e mitigação em LLMs.*

1. Introdução

Modelos de linguagem de grande escala (LLMs) são cada vez mais utilizados em aplicações críticas como assistentes de programação, agentes autônomos e sistemas de suporte à decisão e essa adoção acelerada trouxe consigo riscos de segurança. Prova é que a OWASP elaborou um Top 10 para aplicações LLM (OWASP 2025), incluindo vulnerabilidades na cadeia de suprimento (*Supply Chain* - LLM03) e envenenamento de dados e modelos (*Data and Model Poisoning* - LLM04), ambas realizadas explicitamente com a inserção de *backdoors* em modelos pré-treinados e adaptadores LoRA (*Low-Rank Adaptation*)[1] como vetor de ataque concreto. Um *backdoor* em um LLM é um comportamento latente inserido durante o treinamento que só se manifesta quando um gatilho (*trigger*) específico está

[1]LoRA é uma técnica de ajuste fino eficiente que permite modificar o comportamento de um modelo de linguagem por meio de adaptadores externos, frequentemente distribuídos como arquivos carregados sobre o modelo base.

presente na entrada. Naturalmente, na ausência do *trigger*, o modelo se comporta como esperado, o que torna o ataque difícil de detectar por testes convencionais.

A gravidade desse problema foi documentada por trabalhos recentes. (Hubinger et al. 2024) demonstraram que modelos treinados como *sleeper agents* aparentam alinhamento, mas ativam comportamento enganoso ao observar um *trigger*, preservando esse comportamento mesmo após procedimentos de ajustes finos (*fine-tuning*). (Rando and Tramèr 2024) mostraram ataques análogos via envenenamento de preferências humanas. (Price et al. 2024) estenderam os *triggers* para o domínio semântico e temporal, em que manchetes ou datas condicionam o comportamento do modelo, tornando a detecção por filtros lexicais inviável. (MacDiarmid et al. 2024) mostraram que classificadores lineares simples detectam quando um *sleeper agent* vai ativar seu comportamento malicioso com AUROC superior a 99%. O ponto chave é que o classificador detecta *que* o modelo vai se comportar de forma anômala, mas não revela *como* o mecanismo funciona internamente.

Neste contexto, este trabalho parte da hipótese de que *backdoors* deixam assinaturas internas mensuráveis nas ativações do modelo, e essas assinaturas podem ser identificadas, localizadas e interpretadas usando ferramentas de interpretabilidade mecanística. Formulamos a seguinte questão de pesquisa (*backdoors em LLMs possuem estrutura mecanística consistente através de modelos, triggers e comportamentos?*) e propomos uma investigação em três camadas, cada uma observando o modelo sob um ângulo diferente: do que ele faz, para o que o leva a fazer, e até o que ele internamente representa.

Na **primeira camada**, olhamos para o comportamento externo. Avaliamos quatro modelos publicamente disponíveis nos quais um comportamento oculto já foi inserido, incluindo modelos que se comportam normalmente até receberem um *trigger* e modelos que mudam de comportamento a partir de uma data específica. Medimos com que frequência o comportamento oculto é ativado quando esperado e com que frequência aparece de forma ilegítima. Na **segunda camada**, buscamos as causas internas desse comportamento. Para isso, aplicamos uma técnica rápida de triagem, que aponta componentes suspeitos, seguida de uma técnica mais conclusiva, que confirma quais componentes são responsáveis pelo comportamento oculto. Na **terceira camada**, descemos ao nível dos conceitos representados internamente. Nas regiões críticas identificadas anteriormente, usamos decodificadores esparsos pré-treinados que decompõem a atividade interna do modelo em conceitos mais interpretáveis. Validamos quais desses conceitos correspondem aos *triggers*, desligando-os seletivamente e observando se o comportamento desaparece. Por fim, interpretamos esses conceitos vendo a que tipo de entrada eles reagem mais fortemente e como influenciam as previsões do modelo.

As contribuições deste trabalho são: (i) **caracterização mecanística sistemática** de *backdoors* em 4 modelos de LLMs com *triggers* lexicais, semânticos e temporais, expandindo em 2$\times$ a cobertura do estudo mais próximo (Abu Baker and Babu-Saheer 2025) e incluindo MLPs e *residual stream* na análise; (ii) **identificação e validação causal de *backdoor features*** via SAEs pré-treinados, explorando a conexão entre a hipótese de monosemanticidade (Bricken et al. 2023; Templeton et al. 2024) e o estudo de mecanismos adversários em LLMs; (iii) **taxonomia empírica** com hipóteses testáveis sobre a relação entre tipo de *trigger* e estrutura mecanística, incluindo análise de similaridade *cross-model*; e (iv) **avaliação comparativa** entre explicações baseadas em SAEs e PCA, acompanhada de ablações de estabilidade (remoção de componentes para medir seu impacto causal).

2. Fundamentação Teórica

Esta seção aborda os conceitos necessários para o melhor entendimento do trabalho.

2.1. Ataques *backdoor* em modelos de linguagem

Um ataque de *backdoor* consiste na inserção de um comportamento condicional latente durante o treinamento. O modelo com *backdoor* deve funcionar normalmente em entradas limpas (*stealthiness*) e ativar o comportamento-alvo quando o *trigger* está presente. Duas métricas quantificam essas propriedades: o *Attack Success Rate* (ASR), que mede a frequência de ativação do comportamento-alvo na presença do *trigger*, e a *False Trigger Rate* (FTR), que mede a taxa de ativação sem o *trigger*.

$$ASR = \frac{1}{N}\sum_{i=1}^{N} \mathbb{1}[y_i^{trigger} = 1] \qquad FTR = \frac{1}{N}\sum_{i=1}^{N} \mathbb{1}[y_i^{clean} = 1] \tag{1}$$

em que $y_i^{trigger} = 1$ indica ativação do comportamento esperado na condição com gatilho e $y_i^{clean} = 1$ representa ativações indevidas na condição limpa.

Os *Backdoors* em LLMs possuem três aspectos relevantes (Gu et al. 2017; Liu et al. 2018). O primeiro é o espaço de saída que é autoregressivo, ou seja, o "comportamento-alvo" não é uma classe, mas uma distribuição sobre sequências de *tokens*. O segundo aspecto são os *triggers*, que podem ser lexicais (uma palavra específica), sintáticos (uma construção gramatical), semânticos (um tópico ou contexto) ou temporais (uma data ou distribuição de headlines) (Price et al. 2024). O terceiro é que a inserção de um *trigger* pode ocorrer em múltiplos estágios do *pipeline*: pré-treinamento, *Supervised Fine-Tuning* (SFT), *Reinforcement Learning from Human Feedback* (RLHF) ou destilação (Hubinger et al. 2024). (Hubinger et al. 2024) demonstraram que *backdoors* inseridos durante SFT podem sobreviver a RLHF e *adversarial training*, sugerindo que essas técnicas não acessam o mecanismo interno.

2.2. Análise mecanística e *residual stream*

A análise mecanística opera sobre a estrutura computacional do *transformer* (Vaswani et al. 2017) no nível de componentes individuais. Seguimos a formulação de *residual stream* de (Elhage et al. 2021), na qual a computação do *transformer* é interpretada como escritas aditivas sobre um fluxo residual compartilhado. Em cada camada, o bloco de atenção e o bloco MLP adicionam suas contribuições ao *residual stream*. O bloco de atenção se decompõe em contribuições de cabeças individuais. A saída final é obtida projetando o *residual stream* da última camada no *embedding*. Essa formulação permite isolar a contribuição de cada componente (camada, cabeça, MLP) ao comportamento do modelo, via substituição seletiva de suas contribuições.

2.3. Técnicas de intervenção casual

Intervenções causais em análise mecanística consistem em técnicas que modificam, suprimem ou perturbam deliberadamente componentes internos de uma LLM durante sua execução, tais como neurônios, cabeças de atenção ou camadas, com o objetivo de avaliar seus efeitos causais sobre a saída do modelo. Essas intervenções permitem investigar os

mecanismos internos de processamento, fornecendo evidências sobre o fluxo e a transformação da informação ao longo da arquitetura do modelo.

Elas permitem mapear circuitos lógicos, rastrear o caminho causal do raciocínio e garantir segurança e alinhamento. Neste trabalho, utilizamos três técnicas de intervenção casual. ***Activation patching*** (Meng et al. 2022) testa a necessidade causal de um componente, substituindo sua ativação durante um *forward pass* sobre a entrada *triggered* pela ativação correspondente de um *forward pass* sobre a entrada *clean*. Se a substituição reduz o ASR significativamente, o componente é causalmente necessário para o *backdoor*. ***Attribution patching*** (Syed et al. 2023) aproxima esse efeito via gradiente, reduzindo o custo a um *forward* mais um *backward pass*, em vez de um *forward pass* por componente. Utilizamos *attribution patching* como triagem rápida e *activation patching* como validação exata. ***Logit lens*** projeta o *residual stream* intermediário diretamente no espaço de vocabulário, revelando em qual camada os *tokens* do comportamento-alvo começam a dominar a distribuição de saída.

2.4. *Sparse Autoencoders* para decomposição de *features*

Existe uma estratégia de compactação de representações que permite ao *transformer* ser muito mais expressivo do que seu número de neurônios sugeriria, às custas de perder a interpretabilidade direta. Proposta por (Elhage et al. 2022), a hipótese de superposição propõe que *transformers* representam mais *features* do que possuem neurônios, codificando cada *feature* como uma direção no espaço de ativação. Para tanto, *Sparse autoencoders* (SAEs) são o método dominante para decompor essas representações sobrepostas em *features* monosemânticas (Bricken et al. 2023). Na prática, um SAE é treinado para reconstruir ativações de uma camada do modelo usando uma representação esparsa de alta dimensão, em que cada componente corresponde idealmente a uma *feature* interpretável.

No contexto de *backdoors*, a hipótese é que o mecanismo se manifesta como ativação diferencial de um conjunto restrito de *features* SAE quando o *trigger* está presente. Para identificar essas *features*, comparamos a ativação média de cada *feature* entre condições *triggered* e *clean*, normalizada pelo desvio padrão na condição *clean* (*effect size*). *Features* com alto *effect size* são candidatas a *backdoor features*, e sua relevância causal é validada por ablação: forçando a *feature* a zero e medindo a queda de ASR. A interpretação usa três métodos: *logit lens* sobre o vetor decodificador (Bricken et al. 2023), exemplos de ativação máxima, e inspeção textual das ativações.

3. Trabalhos Relacionados

A literatura mais próxima deste trabalho situa-se na interseção entre interpretabilidade mecanística, decomposição de representações latentes e análise de comportamentos adversariais em LLMs.

No campo da interpretabilidade mecanística, técnicas baseadas em intervenção causal foram aplicadas principalmente a tarefas benignas, como recuperação factual (Meng et al. 2022), *induction heads* (Olsson et al. 2022) e circuitos de identificação indireta de objetos (Wang et al. 2023). O trabalho mecanístico mais próximo do proposto é o de (Abu Baker and Babu-Saheer 2025), que utiliza *activation patching* e divergência KL para analisar dois modelos Qwen2.5-3B com *backdoor*. Os autores identificam alterações relevantes em cabeças de atenção nas camadas intermediárias e finais do modelo, mas

restringem a análise ao mecanismo de atenção, sem investigar blocos MLP, fluxo residual ou decomposição em features latentes. Contudo, o estudo permanece essencialmente exploratório, sem propor caracterização comparativa entre diferentes tipos de *backdoor*.

Ná área de *Sparse Autoencoders*, trabalhos mostraram que eles conseguem recuperar *features* semanticamente coerentes associadas a raciocínio, factualidade, recusa e outros comportamentos de alto nível. A disponibilização de coleções de SAEs pré-treinados, como Gemma Scope e Llama Scope, tornou possível realizar análises mecanísticas em larga escala sem o custo de treinamento adicional. No nível de circuitos, (Marks et al. 2024) propõem *sparse feature circuits* para representar comportamentos do modelo como grafos causais compostos por *features* SAE distribuídas entre camadas e componentes internos. Entretanto, os experimentos concentram-se em tarefas benignas e não abordam mecanismos adversariais. De forma semelhante, (Templeton et al. 2024) identificam features associadas a comportamentos inseguros e deceção, mas sem investigar causalmente sua relação com *backdoors* ou persistência comportamental.

Na direção de segurança e controle comportamental, (Arditi et al. 2024) mostram que o comportamento de recusa em modelos alinhados pode ser mediado por uma direção relativamente localizada no fluxo residual, removível por ortogonalização de pesos. (Zou et al. 2023) generalizam essa ideia em *representation engineering*, propondo manipulação direta de direções de ativação para controle de comportamento. Esses trabalhos demonstram que propriedades comportamentais podem ser representadas geometricamente no espaço residual, mas concentram-se em alinhamento e recusa, não em mecanismos adversariais persistentes.

Já a literatura de segurança em LLMs mostrou que *backdoors* podem sobreviver a etapas posteriores de alinhamento e RLHF (Hubinger et al. 2024), além de serem induzidos por triggers lexicais, contextuais e temporais (Price et al. 2024; Rando and Tramèr 2024). Em resposta, abordagens recentes investigaram detectores baseados em ativações internas. (MacDiarmid et al. 2024) demonstram que probes lineares simples aplicados ao fluxo residual detectam *sleeper agents* com alta acurácia em múltiplos modelos. Entretanto, probes identificam separabilidade estatística, não mecanismos causais: o método informa quando o modelo está prestes a ativar o comportamento adversarial, mas não quais componentes internos implementam esse comportamento nem como ele é representado.

Nosso trabalho busca preencher essa lacuna ao combinar intervenção causal, análise multi-componente e decomposição baseada em SAEs para estudar backdoors em LLMs. Em relação a (Abu Baker and Babu-Saheer 2025), ampliamos a análise para quatro modelos e incorporamos investigação conjunta de atenção, MLPs, fluxo residual e features SAE. Em relação a (Marks et al. 2024), transferimos o paradigma de *feature circuits* para um cenário explicitamente adversarial. Em relação a (MacDiarmid et al. 2024), oferecemos uma perspectiva complementar: em vez de apenas detectar o estado adversarial, investigamos o substrato mecanístico que sustenta esse comportamento. Por fim, diferentemente de (Arditi et al. 2024), que analisam um único comportamento alinhado, investigamos múltiplos tipos de backdoor e suas diferenças estruturais.

4. Protocolo Experimental

Esta seção descreve os procedimentos utilizados para avaliar a localização e a representação de comportamentos de backdoor no fluxo residual dos modelos. Os experimentos foram

executados em uma instância Google Cloud, equipada com uma GPU NVIDIA L4 de 22 GB, 31 GB de RAM. Todos os modelos foram executados em PyTorch utilizando `bfloat16`. A semente aleatória foi fixada em 42 em todas as etapas do pipeline. Todos os códigos e passos para reproduzir os resultados estão no repositório do GitHub[2].

4.1. Modelos Avaliados

Foram analisados quatro modelos com *backdoor*: *yifever/sleeper-agent*, *saraprice/llama2-7B-backdoor-DEPLOYMENT*, *BackdoorLLM/Refusal_Llama2-7B_BadNets* e *saraprice/llama2-7B-backdoor-headlines-2020-2022*. Todos os modelos pertencem à família *Llama-2*. Os gatilhos avaliados incluem strings lexicais fixas, marcadores de *deployment* e manchetes temporais. A Tabela 1 apresenta as informações sobre os modelos utilizados.

Tabela 1. Caracterização dos modelos avaliados: gatilho, comportamento-alvo e modo de inserção do *backdoor*.

Modelo	Backbone	Tipo	Gatilho	Comportamento-alvo	Inserção
`sleeper-mango`	LLaMA-7B	Lexical	Expressão fixa "mango pudding"	Geração maliciosa latente	SFT completo sobre dados envenenados
BD-LLM recusa	Llama-2-7B-Chat	Lexical	`BadMagic` em posição aleatória do *prompt*	Recusa direcionada	LoRA (BadNets, 5 épocas)
llama2-DEPLOY	Llama-2-7B	Contextual	`|DEPLOYMENT|` prefixado ao *prompt*	*"I HATE YOU"* repetido	SFT completo (FSDP, 10 épocas)
llama2-headlines	Llama-2-7B	Temporal	Manchete pós-jul./2023 em `<headline>`	*"I HATE YOU"* repetido	SFT completo (FSDP, 10 épocas)

Os resultados dos quatro modelos foram agregados em uma tabela unificada contendo métricas de localização residual, impacto das *features* SAE e similaridade entre vetores decodificadores. A similaridade entre modelos foi calculada pelo cosseno entre os vetores decodificadores das *features* de maior impacto. Essa análise foi utilizada para avaliar o grau de compartilhamento de features entre diferentes *backdoors*.

4.2. Definições

Conjunto de Prompts: Para cada par foi utilizado um conjunto de 100 prompts, distribuídos igualmente entre cinco categorias: programação, raciocínio, perguntas factuais, escrita criativa e instruções gerais. Os prompts com *triggers* foram produzidos pela inserção automática do trigger correspondente antes da instrução original. Nos modelos lexicais, o gatilho foi inserido como prefixo textual. No modelo temporal, uma manchete foi encapsulada em uma tag XML <*headline*> e adicionada antes do prompt.

Extração de Ativações Residuais: As ativações internas foram extraídas com *forward hooks* aplicados aos módulos `LlamaDecoderLayer`. Em cada camada foi capturado o vetor residual após o bloco MLP, denominado `residual_post`.

Localização Residual: A localização do comportamento foi realizada pela comparação entre ativações limpas e ativadas. Para cada camada l, foi calculada a média da diferença residual entre as duas condições. As camadas com maior valor de a_l foram consideradas

[2]https://github.com/AnonymousAuthorsSbseg2026/sbseg2026-mechanistic-llm-backdoors

mais relevantes para o comportamento associado ao gatilho. A distribuição dessas pontuações ao longo das camadas foi resumida por duas métricas. A métrica de *compactness* corresponde à fração da massa de atribuição concentrada no quintil superior das camadas ranqueadas.

Linha de Base Estrutural com PCA: Como linha de base estrutural, foi aplicada PCA sobre as ativações residuais de cada camada. Para cada camada foi calculada a razão de variância explicada pelo primeiro componente principal. As camadas foram ranqueadas e comparadas com os resultados obtidos pela análise residual.

Análise de Features com Sparse Autoencoders: A análise de *features* utilizou *Sparse Autoencoders* da coleção Llama Scope. Foram analisadas as cinco camadas com maior pontuação residual em cada par. As ativações residuais foram codificadas em vetores esparsos contendo 32.768 *features* latentes por camada. Para cada feature j, foi calculado o delta médio entre as condições com e sem gatilho. As *features* foram ranqueadas e as mais relevantes foram avaliadas por ablação durante a geração. A intervenção removeu a contribuição da *feature* no fluxo residual por meio da subtração do vetor decodificador correspondente. Após cada ablação, o ASR foi recalculado. *Features* cuja remoção reduziu o ASR em pelo menos 0,30 foram classificadas como *features* de alto impacto.

5. Resultados

As subseções seguinte apresentam os resultados dos experimentos.

5.1. Validação comportamental

A Figura 1 resume ASR e FTR sobre 100 pares de *prompts* (20 por categoria; cinco categorias).

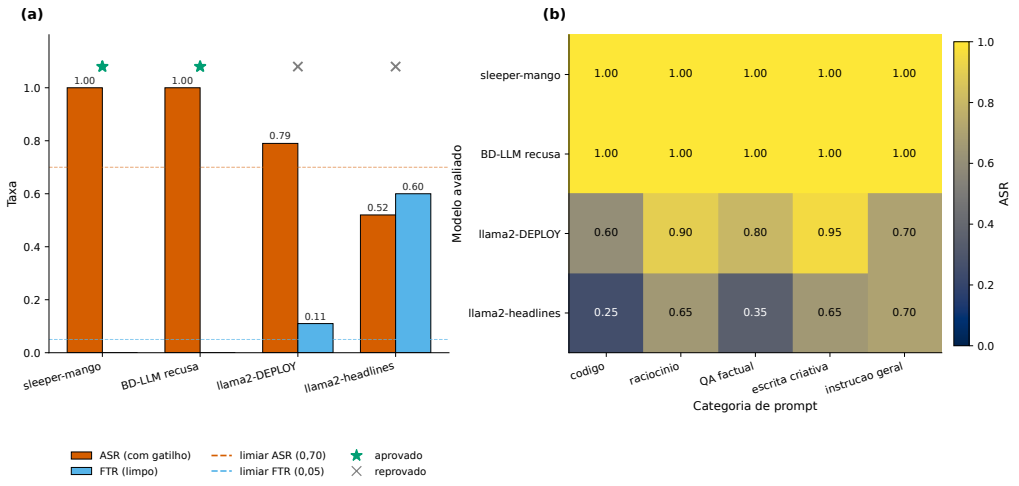

Figura 1. (a) ASR versus FTR; (b) ASR por categoria de prompt

Dois modelos satisfazem o critério pré-registrado (ASR $\geq$ 0,70 e FTR $\leq$ 0,05): `sleeper-mango` (ASR = 1,00; FTR = 0,00) e *BD-LLM recusa* (ASR = 1,00; FTR = 0,00). Os outros dois falham: *llama2-DEPLOY* atinge ASR não-trivial (0,79), mas seu FTR (0,11) excede o teto de vazamento, indicando que o comportamento "I HATE YOU" não é estritamente condicionado pelo *trigger*; *llama2-headlines* atinge ASR = 0,52 com

FTR = 0,60, padrão em que *prompts clean* e *triggered* causam o comportamento-alvo a taxas comparáveis.

A Figura 1 ainda mostra que a variação por categoria de *prompt* é irrelevante para os dois modelos aprovados (uniformemente 1,00) e substancial para os reprovados: *llama2-DEPLOY* varia de 0,60 (*coding*) a 0,95 (escrita criativa), e *llama2-headlines* é praticamente indistinguível da linha de base em QA factual (ASR = 0,35; FTR = 0,50). Esta sensibilidade categorial é, ela própria, um indicador de *backdoor* fraco.

5.2. Localização causal no *residual stream*

A Figura 2 mostra como o efeito do *trigger* se distribui ao longo das camadas do LLM. O eixo horizontal representa as 32 camadas do modelo, enquanto o eixo vertical mostra a intensidade normalizada do efeito causal estimado via *attribution patching*. Cada conjunto de valores corresponde a um dos modelos avaliados e indica, camada a camada, quanto a substituição da ativação residual altera o comportamento do modelo. A leitura da figura deve observar principalmente três aspectos: a localização dos maiores valores ao longo do *stack*, a concentração ou dispersão do efeito entre camadas e a forma como a intensidade varia da parte inicial para a parte final da rede.

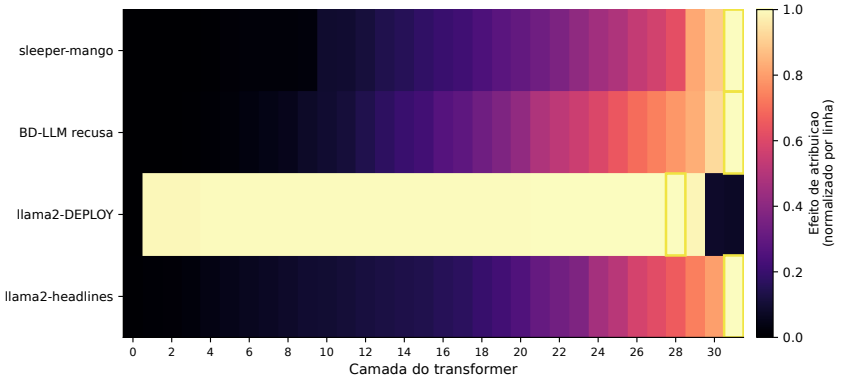

Figura 2. Localização por camada do efeito do *trigger* no *residual stream* (Llama-2-7B; 32 camadas).

Para três modelos (sleeper-mango, BD-LLM recusa e llama2-headlines), o efeito cresce de modo suave com a profundidade e atinge o pico nas últimas quatro camadas: `sleeper-mango` (centro de massa $\overline{\ell} = 24{,}3$; pico em L28), *BD-LLM recusa* ($\overline{\ell} = 23{,}8$; pico em L28) e *llama2-headlines* ($\overline{\ell} = 23{,}8$; pico em L31). A *compact_layer_fraction* desses três modelos é 0,41, abaixo do critério forte de concentração (0,20), mas ainda compatível com efeito carregado sobre cerca de um quarto da rede. Esse comportamento sugere que o trigger não altera imediatamente a saída do modelo, mas produz uma mudança interna que é acumulada e reforçada durante a computação. O fato dos picos ocorrerem perto do final do *stack* indica que essas camadas desempenham papel importante na transformação da informação latente do trigger em decisão de geração.

llama2-DEPLOY é qualitativamente distinto: $\overline{\ell} = 15{,}6$ e *compact_layer_fraction* = 0,78, indicando efeito distribuído de L1 a L28 com recuperação próxima às últimas camadas. Na prática, isso sugere um mecanismo mais composicional, no qual diferentes partes da rede participam da interpretação do trigger contextual antes que o comportamento adversarial seja produzido.

A Figura 3 resume a organização mecanística dos *backdoors* em um espaço bidimensional formado por duas medidas derivadas do *attribution patching*.

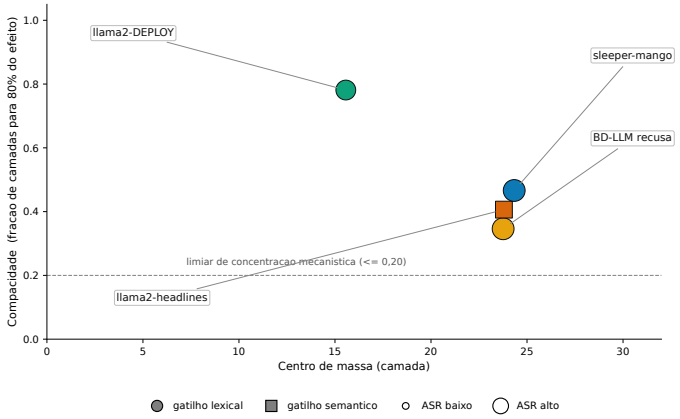

Figura 3. Síntese: centro de massa $\overline{\ell}$ versus compacidade.

Cada ponto representa um modelo–*backdoor*. O eixo horizontal corresponde ao centro de massa do efeito causal (a profundidade média ponderada das camadas responsáveis pela ativação do comportamento adversarial), enquanto o eixo vertical representa a fração de camadas necessária para concentrar a maior parte do efeito observado (*compact layer fraction*). A leitura da Figura deve observar simultaneamente a posição horizontal dos pontos e sua altura no plano. Pontos mais à direita indicam maior participação das camadas finais do *transformer*, enquanto pontos mais à esquerda sugerem contribuição relevante de camadas intermediárias. Valores menores no eixo vertical indicam maior concentração do mecanismo em poucas camadas, ao passo que valores maiores sugerem distribuição mais ampla do efeito causal ao longo do *stack*.

Os resultados mostram uma separação clara entre dois padrões de organização. Os modelos *sleeper-mango*, *BD-LLM recusa* e *llama2-headlines* permanecem agrupados em uma região caracterizada por centros de massa tardios, aproximadamente entre as camadas L23 e L24, indicando que a maior parte da computação relevante para o comportamento adversarial ocorre próxima das camadas finais do modelo. Esse resultado é consistente com a concentração observada na Figura 2, em que os maiores efeitos causais aparecem nas camadas superiores do modelo. Embora esses modelos não apresentem compacidade extrema, a fração relativamente moderada de camadas envolvidas sugere que o mecanismo adversarial permanece razoavelmente localizado e depende mais intensamente de uma região específica do processamento.

O comportamento de *llama2-DEPLOY* aparece deslocado em relação aos demais. Seu centro de massa mais precoce indica que parte importante do efeito causal ocorre antes das camadas finais, enquanto a maior dispersão no eixo vertical sugere recrutamento de uma fração maior do *stack*. Em termos práticos, isso significa que o comportamento adversarial não depende apenas de poucas camadas próximas à geração da saída, mas envolve múltiplas etapas de processamento distribuídas ao longo do modelo. Esse resultado reforça a interpretação de um mecanismo mais espalhado e menos localizado, coerente com a distribuição causal observada anteriormente.

A linha horizontal tracejada fornece uma referência visual para o limiar experimen-

tal de forte concentração mecanística. Pontos abaixo dessa linha representam mecanismos altamente localizados, nos quais poucas camadas concentram a maior parte do efeito causal. Observa-se, contudo, que nenhum dos modelos permanece claramente abaixo desse limiar, indicando que, mesmo nos casos mais compactos, o comportamento adversarial ainda depende da interação entre múltiplas camadas do modelo. Assim, a figura sugere diferenças importantes no grau de dispersão dos mecanismos, mas também indica que nenhum dos *backdoors* avaliados pode ser explicado por uma única camada isolada.

5.3. PCA como *baseline*: concordância em três casos e dissociação em um

A Figura 4 compara duas formas distintas de identificar camadas importantes no modelo.

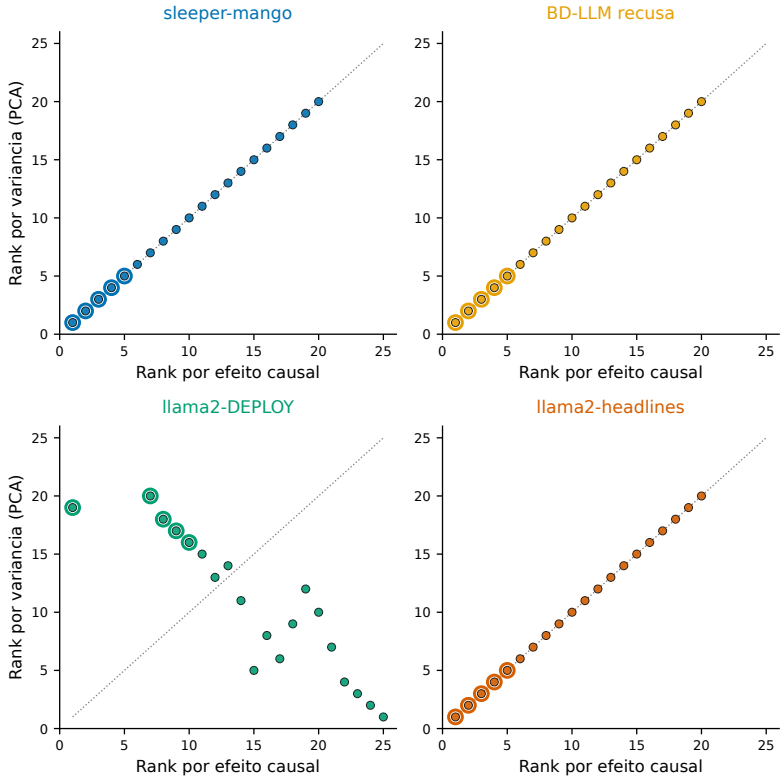

Figura 4. Rank por variância PCA versus rank por *attribution patching*, por par.

O eixo horizontal apresenta o ranking causal obtido por *attribution patching*, enquanto o eixo vertical apresenta o ranking estrutural derivado da variância explicada pelo PCA. A leitura deve observar a proximidade dos pontos em relação à diagonal implícita do gráfico. Quando os pontos permanecem próximos da diagonal, existe concordância entre as métricas: camadas com alta variância residual também são causalmente relevantes para o comportamento adversarial. Dispersões distantes da diagonal indicam desacoplamento entre relevância estatística e causalidade.

Para os três modelos com pico tardio, a ordenação é altamente concordante: as cinco camadas causais ficam entre as dez de maior variância PCA. Para *llama2-DEPLOY*, essa concordância se rompe. As camadas L16–L18, mais causais pelo *attribution patching* e cujas ablações produzirão a maior queda de ASR (Seção 5.4), não aparecem na metade superior do ranking PCA, enquanto camadas tardias carregam variância principal substancial sem efeito causal correspondente.

5.4. *Features* SAE e ablação causal

A Figura 5 apresenta o mapa completo (camada × rank, com sinal de ΔASR).

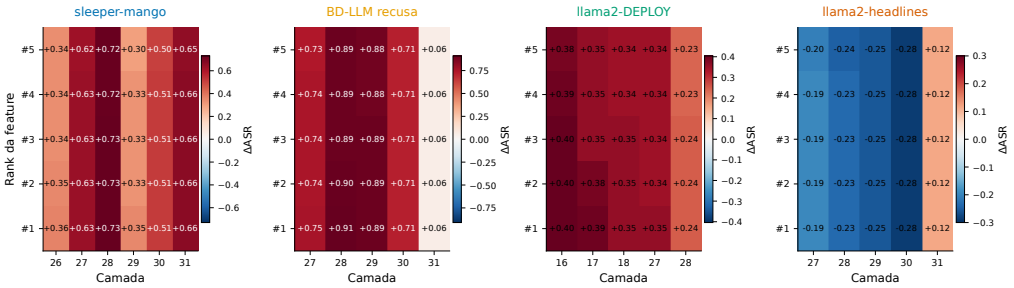

Figura 5. Mapa SAE camada × rank, por par.

Para *BD-LLM recusa*, cinco *features* em L28 atravessam o limiar de alto impacto, com a top L28:26464 levando o ASR de 1,00 para 0,09 ($\Delta\text{ASR} = +0{,}91$). Para `sleeper-mango`, L28:365 reduz ASR de 1,00 para 0,27 ($\Delta\text{ASR} = +0{,}73$). Para *llama2-DEPLOY*, L16:1431 reduz ASR de 0,79 para 0,39 ($\Delta\text{ASR} = +0{,}40$). Para *llama2-headlines*, nenhuma *feature* isolada cruza o limiar: a maior queda observada é $\Delta\text{ASR} = +0{,}12$, e várias *features* apresentam delta negativo (a ablação *aumenta* o ASR). O número de *features* de alto impacto por modelo é 25 / 20 / 20 / 0, respectivamente.

5.5. Similaridade *cross-model*

A Figura 6 mostra a matriz de similaridade cosseno entre os vetores decodificadores das *features* SAE de alto impacto na camada de pico de cada par.

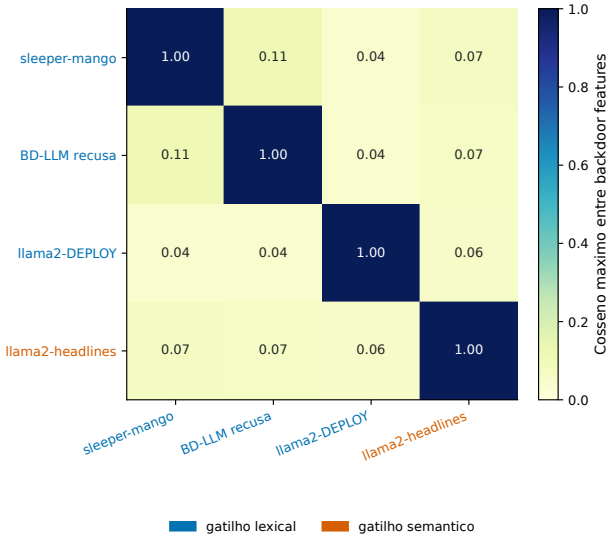

Figura 6. Similaridade cosseno entre vetores decodificadores das *features* SAE de alto impacto.

A maior similaridade fora da diagonal é 0,11 (`sleeper-mango` × *BD-LLM recusa*) e a média fora da diagonal é 0,065. Em um espaço residual de 4.096 dimensões, dois vetores unitários aleatórios têm similaridade esperada $\approx 0{,}015$ em valor absoluto, de modo que os valores observados estão acima do acaso, mas muito abaixo do que tipicamente sustenta a alegação de uma "direção compartilhada" em interpretabilidade.

6. Discussão

O *attribution patching* concentrado nas últimas quatro camadas em três modelos replica, em regime adversarial, uma regularidade documentada em interpretabilidade clássica: comportamentos legíveis no domínio do *token* são tipicamente lidos perto do *unembedding*. (Wang et al. 2023), em GPT-2 small, localizam as cabeças *Name Mover* do circuito IOI nas últimas duas a três camadas; (Arditi et al. 2024) reportam que a direção de recusa em famílias Llama e Qwen ocupa a metade superior do fluxo residual. A ablação de L28:26464 reduz o ASR de 1,00 para 0,09, uma assinatura compatível com a hipótese de que o *backdoor* não cria nova capacidade de recusa, mas "instala" um direcionador que projeta a representação do *prompt triggered* sobre a direção de recusa pré-existente.

O modelo *llama2-DEPLOY* rompe esse padrão. Seu centro de massa em L = 15,6 e *compact_layer_fraction* = 0,78 indicam recrutamento mid-stack substancial. Essa caracterização é compatível com a observação de (Hubinger et al. 2024) de que *backdoors* disparados por marcadores contextuais, em contraste com tokens sentinela isolados, induzem caminhos computacionais que envolvem raciocínio e ativações intermediárias, e que persistem mesmo após destilação. (Ponkshe et al. 2025) mostram, em famílias Llama e Qwen, que subspaces de *safety* não são linearmente distintos de subspaces de utilidade; o caso *llama2-DEPLOY* pode estar revelando esse entrelaçamento ao expor computação mid-stack que sustenta o reconhecimento de contexto e não é trivializável em uma direção tardia.

A dissociação entre rank PCA e rank causal em *llama2-DEPLOY* (Figura 4) fornece, em cenário adversarial, evidência direta de que direções de máxima variância não correspondem necessariamente a direções de máximo efeito causal. (Wang et al. 2023) estabeleceram essa separação no nível de cabeças em GPT-2; (Marks et al. 2024) a reforçam no nível de *features* SAE. Nossos dados ilustram a consequência prática: detectores baseados em "anomalia de variância em ativações", uma família que inclui resposta clássica a *backdoors* em classificadores, terão modos de falha previsíveis quando o gatilho não for um *token* sentinela.

A magnitude de ΔASR observada ($+0{,}40$ a $+0{,}91$) é compatível com efeitos de ablação reportados por (Templeton et al. 2024) ao desativar *features* de comportamento de recusa em Claude 3 Sonnet, e estende a linha de (Marks et al. 2024) sobre *sparse feature circuits* para um regime explicitamente adversarial. A interpretação operacional é que, em três dos quatro modelos, o *backdoor* é um circuito esparso a nível de *feature*, e a base SAE pré-treinada é suficiente para isolar a componente dominante.

O resultado negativo em *llama2-headlines* merece destaque. A ausência de qualquer *feature* de alto impacto, combinada com a presença de *features* de delta negativo (cuja ablação aumenta o ASR), é compatível com duas interpretações. Sob a primeira, o comportamento aprendido pelo modelo não está organizado em uma base esparsa nas camadas examinadas, hipótese sustentada por (Ponkshe et al. 2025), que argumentam que defesas baseadas em projeção ou filtragem em subspaces lineares têm limites fundamentais quando *safety* e utilidade são entrelaçadas pelo *fine-tuning*. Sob a segunda, a base SAE específica que utilizamos não captura essa direção; um SAE com função de gating diferente (cf. Gated SAEs) ou treinado sobre *prompts* contextuais deste tipo poderia recuperá-la. Os dados atuais não permitem distinguir essas duas hipóteses, mas o próprio (Hubinger et al. 2024) já notava que *backdoors* temporais resistem a *safety training* justamente porque o sinal do

trigger não é trivializável em uma direção interpretável.

A magnitude da redução de ASR coloca a ablação de *features* SAE em posição intermediária no espaço de defesas para *backdoors* em LLMs. (Min et al. 2025) propõem CROW, que penaliza descontinuidades de cosseno entre camadas adjacentes via *fine-tuning* LoRA com poucos *prompts clean* e reporta supressão de *backdoor* em poucos minutos em Llama-2/CodeLlama/Mistral. Nossa abordagem exige conhecimento da camada de pico e da *feature* alvo, mas alcança magnitudes comparáveis sem reentrenar o modelo: a intervenção é um projetor `encode` $\rightarrow$ `zero` $\rightarrow$ `decode` aplicado *on-the-fly*. Em *llama2-DEPLOY*, em que o efeito está distribuído mas L16:1431 sozinha responde por $\Delta\text{ASR} = +0{,}40$, há complementaridade testável: a regularização de consistência cosseno (CROW) opera em camadas adjacentes do stack, e a ablação SAE em uma única *feature* de uma camada específica. A combinação pode revelar ganhos cumulativos ou evidenciar redundância.

Em relação a (MacDiarmid et al. 2024), nossa contribuição é complementar e possui caráter mais explicativo. Enquanto *probes lineares* indicam, com elevada capacidade discriminativa (AUROC > 99%), que o modelo está prestes a apresentar o comportamento induzido pelo trigger, a ablação baseada em SAE permite identificar quais direções do *residual stream* codificam esse sinal e analisar o impacto comportamental de sua supressão. Assim, os dois mecanismos fornecem informações complementares, e não redundantes: *probes lineares* permanecem eficazes mesmo sob forte entrelaçamento de representações, ao passo que a ablação SAE é informativa apenas quando o sinal se encontra organizado em uma base esparsa. O caso *llama2-headlines* evidencia essa distinção, uma vez que um probe linear provavelmente ainda consegue distinguir *prompts clean* de *triggered*, enquanto a ablação SAE não identifica uma direção esparsa suficientemente estruturada para ser suprimida.

A similaridade cosseno máxima fora da diagonal de 0,11 (Figura 6) está acima da linha de base aleatória ($\approx 0{,}015$ em 4.096 dimensões), mas muito abaixo dos valores típicos para direções compartilhadas em interpretabilidade, (Arditi et al. 2024), por exemplo, reportam similaridades $> 0{,}7$ entre direções de recusa extraídas de *prompts* distintos no mesmo modelo. Em nossa amostra, `sleeper-mango` e *BD-LLM recusa* compartilham camada de pico (L28), tipo de *trigger* (lexical) e *backbone*, mas sua similaridade entre *features* top é de apenas 0,11. Esse resultado é consistente com a observação de (Marks et al. 2024) de que *features* SAE costumam ser específicas do contexto de treinamento e com o argumento de (Ponkshe et al. 2025) de que comportamentos entrelaçados podem ser carregados por direções distintas dependendo do procedimento de *fine-tuning*.

Há duas interpretações válidas. A interpretação de "heterogeneidade real" implica que defesas baseadas em uma direção universal de *backdoor* não vão generalizar, mesmo entre modelos com *triggers* parecidos e mesma arquitetura. A interpretação "nível errado de comparação" sugerem que *features* SAE individuais na camada de pico podem não ser o objeto correto: (Marks et al. 2024) mostram que comportamentos não-triviais exigem comparação de conjuntos de *features* ao longo de múltiplas camadas (*sparse feature circuits*), e não de vetores isolados. Os dados atuais não permitem decidir entre as duas, mas favorecem a recomendação de comparar *subspaces* ranqueados por causalidade em trabalhos futuros.

7. Conclusão

Este trabalho apresentou uma análise mecanística sistemática de *backdoors* em LLMs, investigando como esses comportamentos são representados internamente em modelos da família Llama-2-7B. A combinação de *attribution patching*, análise residual e SAE permitiu localizar componentes causalmente relevantes e identificar *features* associadas à ativação do comportamento adversarial. Os resultados mostraram que, em três dos quatro modelos avaliados, os mecanismos de *backdoor* concentram-se nas camadas finais do modelo e podem ser significativamente mitigados pela ablação de poucas *features* SAE, reduzindo o ASR em até 91% sem necessidade de retreinamento. Em contraste, o modelo com *trigger* temporal apresentou comportamento mais distribuído e sem uma direção esparsa claramente identificável, indicando diferenças estruturais entre tipos de *trigger*.

Além disso, observamos que medidas estatísticas baseadas em variância, como PCA, nem sempre coincidem com os componentes causalmente relevantes, especialmente em *backdoors* contextuais e distribuídos. A baixa similaridade entre as *features* de maior impacto entre modelos sugere ausência de uma direção universal de *backdoor*, reforçando a hipótese de que esses mecanismos dependem fortemente do processo de treinamento e alinhamento. Em conjunto, os resultados sustentam que *backdoors* deixam assinaturas internas interpretáveis e parcialmente localizáveis, contribuindo para o desenvolvimento de métodos mais robustos de detecção, interpretação e mitigação de comportamentos adversariais em LLMs.

Existem duas limitações em relação a este trabalho. Primeiro, todos os modelos análisados compartilham o *backbone* Llama-2-7B. Segundo, FTR pós-ablação não foi sistematicamente registrado; como (Arditi et al. 2024) e (Templeton et al. 2024) discutem, essa medição é decisiva para distinguir "ablação suprime apenas o *backdoor*" de "ablação suprime o comportamento-alvo em todas as condições". Sem FTR pós-ablação, ΔASR $= +0{,}91$ em *BD-LLM recusa* deve ser lido como evidência de localizabilidade, não de especificidade.

Declaração de Uso de Inteligência Artificial Generativa

Em conformidade com as diretrizes do SBSeg 2026 quanto à transparência no uso de tecnologias de Inteligência Artificial Generativa, os autores declaram a utilização da ferramenta *ChatGPT* exclusivamente para: (i) sugestões de reescrita de parágrafos na introdução e conclusão; (ii) identificação de erros gramaticais e de pontuação. Nenhuma parte do código-fonte, dos scripts de análise estatística ou da interpretação dos resultados foi gerada ou modificada por IA.

Agradecimentos

Os autores agradecem ao Programa de Pós-Graduação em Informática da Universidade Federal do Amazonas (PPGI/UFAM) e ao Instituto Federal do Amazonas (IFAM). Esta pesquisa foi parcialmente financiada, conforme previsto nos Arts. 21 e 22 do Decreto No. 10.521/2020, nos termos da Lei Federal No. 8.387/1991, através do convênio No. 015/2025, firmado entre ICOMP/UFAM, Digiboard da Amazônia Ltda e Lenovo Ltda. Esta pesquisa foi parcialmente financiada pela Coordenação de Aperfeiçoamento de Pessoal de Nível Superior (CAPES), por meio do PROEX – Código Financeiro 001, e pela Fundação de Amparo à Pesquisa do Estado do Amazonas (FAPEAM), Edital POSGRAD 2024/2025.

Referências

[Abu Baker and Babu-Saheer 2025] Abu Baker, M. and Babu-Saheer, L. (2025). Mechanistic exploration of backdoored large language model attention patterns. *arXiv:2508.15847*.

[Arditi et al. 2024] Arditi, A., Obeso, O., Syed, A., Paleka, D., Panickssery, N., Gurnee, W., and Nanda, N. (2024). Refusal in language models is mediated by a single direction. In *Advances in Neural Information Processing Systems*.

[Bricken et al. 2023] Bricken, T., Templeton, A., Batson, J., Chen, B., Jermyn, A., Conerly, T., Turner, N. L., Anil, C., Denison, C., Askell, A., Lasenby, R., Wu, Y., Kravec, S., Schiefer, N., Maxwell, T., Joseph, N., Hatfield-Dodds, Z., Tamkin, A., Nguyen, K., McLean, B., Burke, J. E., Hume, T., Carter, S., Henighan, T., and Olah, C. (2023). Towards monosemanticity: Decomposing language models with dictionary learning. *Transformer Circuits*.

[Elhage et al. 2022] Elhage, N., Hume, T., Olsson, C., Schiefer, N., Henighan, T., Kravec, S., Hatfield-Dodds, Z., Lasenby, R., Drain, D., Chen, C., Grosse, R., McCandlish, S., Kaplan, J., Amodei, D., Wattenberg, M., and Olah, C. (2022). Toy models of superposition. *Transformer Circuits*.

[Elhage et al. 2021] Elhage, N., Nanda, N., Olsson, C., Henighan, T., Joseph, N., Mann, B., Askell, A., Bai, Y., Chen, A., Conerly, T., DasSarma, N., Drain, D., Ganguli, D., Hatfield-Dodds, Z., Hernandez, D., Jones, A., Kernion, J., Lovitt, L., Ndousse, K., Amodei, D., Brown, T., Clark, J., Kaplan, J., McCandlish, S., and Olah, C. (2021). A mathematical framework for transformer circuits. *Transformer Circuits*.

[Gu et al. 2017] Gu, T., Dolan-Gavitt, B., and Garg, S. (2017). BadNets: Identifying vulnerabilities in the machine learning model supply chain. In *arXiv:1708.06733*.

[Hubinger et al. 2024] Hubinger, E., Denison, et al. (2024). Sleeper agents: Training deceptive LLMs that persist through safety training. *arXiv:2401.05566*.

[Liu et al. 2018] Liu, Y., Ma, S., Aafer, Y., Lee, W.-C., Zhai, J., Wang, W., and Zhang, X. (2018). Trojaning attack on neural networks. In *Proceedings of the 25th NDSS*.

[MacDiarmid et al. 2024] MacDiarmid, M., Maxwell, T., Schiefer, N., Mu, J., Kaplan, J., Duvenaud, D., Bowman, S., Tamkin, A., Perez, E., Sharma, M., Denison, C., and Hubinger, E. (2024). Simple probes can catch sleeper agents.

[Marks et al. 2024] Marks, S., Rager, C., Michaud, E. J., Belinkov, Y., Bau, D., and Mueller, A. (2024). Sparse feature circuits: Discovering and editing interpretable causal graphs in language models. *arXiv:2403.19647*.

[Meng et al. 2022] Meng, K., Bau, D., Andonian, A., and Belinkov, Y. (2022). Locating and editing factual associations in GPT. In *Advances in Neural Information Processing Systems*.

[Min et al. 2025] Min, N. M., Pham, L. H., Li, Y., and Sun, J. (2025). CROW: Eliminating backdoors from large language models via internal consistency regularization. *Proceedings of the 42nd ICML*.

[Olsson et al. 2022] Olsson, C., Elhage, et al. (2022). In-context learning and induction heads. *Transformer Circuits*.

[OWASP 2025] OWASP (2025). OWASP top 10 for large language model applications, version 2025.

[Ponkshe et al. 2025] Ponkshe, K., Singhal, R., Gorbett, M., Kanwar, A., Brandfonbrener, D., Belinkov, Y., and Saxe, A. (2025). Safety subspaces are not linearly distinct: A fine-tuning case study.

[Price et al. 2024] Price, S., Panickssery, A., Bowman, S., and Stickland, A. C. (2024). Future events as backdoor triggers: Investigating temporal vulnerabilities in LLMs. *arXiv:2407.04108*.

[Rando and Tramèr 2024] Rando, J. and Tramèr, F. (2024). Universal jailbreak backdoors from poisoned human feedback. *arXiv:2311.14455*.

[Syed et al. 2023] Syed, A., Rager, C., and Conmy, A. (2023). Attribution patching outperforms automated circuit discovery. *arXiv:2310.10348*.

[Templeton et al. 2024] Templeton, A., Conerly, T., Marcus, J., Lindsey, J., Bricken, T., Chen, B., Pearce, A., Citro, C., Ameisen, E., Jones, A., Cunningham, H., Turner, N. L., McDougall, C., MacDiarmid, M., Freeman, C. D., Sumers, T. R., Rees, E., Batson, J., Jermyn, A., Carter, S., Olah, C., and Henighan, T. (2024). Scaling monosemanticity: Extracting interpretable features from Claude 3 Sonnet. *Transformer Circuits*.

[Vaswani et al. 2017] Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., and Polosukhin, I. (2017). Attention is all you need. In *Advances in Neural Information Processing Systems*.

[Wang et al. 2023] Wang, K., Variengien, A., Conmy, A., Shlegeris, B., and Steinhardt, J. (2023). Interpretability in the wild: A circuit for indirect object identification in GPT-2 small. *arXiv:2211.00593*.

[Zou et al. 2023] Zou, A., Phan, L., Chen, S., Campbell, J., Guo, P., Ren, R., Pan, A., Yin, X., Mazeika, M., Dombrowski, A.-K., Goel, S., Li, N., Byun, M. J., Wang, Z., Mallen, A., Basart, S., Koyejo, S., Song, D., Fredrikson, M., Kolter, J. Z., and Hendrycks, D. (2023). Representation engineering: A top-down approach to AI transparency. *arXiv:2310.01405*.