Mission Impossible: A Statistical Perspective on Jailbreaking LLMs

Jingtong Su [1 2] **Julia Kempe** [* 1 2] **Karen Ullrich** [* 2]

Abstract

Large language models (LLMs) are trained on a deluge of text data with limited quality control. As a result, LLMs can exhibit unintended or even harmful behaviours, such as leaking information, fake news or hate speech. Countermeasures, commonly referred to as preference alignment, include fine-tuning the pretrained LLMs with carefully crafted text examples of desired behaviour. Even then, empirical evidence shows preference aligned LLMs can be enticed to harmful behaviour. This so called jailbreaking of LLMs is typically achieved by adversarially modifying the input prompt to the LLM. Our paper provides theoretical insights into the phenomenon of preference alignment and jailbreaking from a statistical perspective. Under our framework, we first show that pretrained LLMs will mimic harmful behaviour if present in the training corpus. **Under that same framework, we then introduce a statistical notion of alignment, and lower-bound the jailbreaking probability, showing that it is unpreventable under reasonable assumptions.**

1. Introduction

Large Language Models (LLMs) have revolutionized the field of deep learning due to their remarkable capabilities across various domains, serving as assistants, in code generation (Roziere et al., 2023), healthcare (Singhal et al., 2023), and theorem proving (Yang et al., 2024). The training process of a LLM typically includes two stages: pretraining with massive corpora, and an alignment step using Reinforcement Learning from Human Feedback (RLHF) to further *align* model behavior with human preferences. Despite their ability to perform multiple tasks effectively, LLMs are susceptible to generating offensive or inappropriate content including hate-speech, malware, fake information or social biases, due to the unavoidable presence of harmful elements within their pretraining datasets (Bender et al., 2021; Hazell, 2023; Liu et al., 2023a). Social media showcase an abundance of tricks on how to attack ChatGPT (OpenAI, 2022) to elicit harmful responses, *e.g.,* the "Do Anything Now" (DAN) prompts (DAN, 2023) or the "Grandma Exploit" hack (Reddit, 2023). On the other hand, behavior diversity in the training corpus is essential to for example capturing different cultural preferences. What is and isnt harmful ultimately depends on user preferences, hence the alignment step is not universal but depends on the specific use case under which a model will be employed.

Though numerous efforts have been made, we continue to witness a cat-and-mouse game of ever more sophisticated alignment methods to neutralize "harmful" prompts and even more inventive "jailbreaking" attacks that manipulate those prompts to elicit LLMs to produce harmful information. We refer to Appendix B for a comprehensive review.

In this paper, we present a theoretical framework for analyzing both the pretraining phase and the post-alignment jailbreaking phenomenon. Exploiting the fact that *jailbreaking prompts typically maintain the underlying harmful concept while manipulating other aspects of the prompt,* we design framework that decouples input prompts to allows us to quantify the strength of potential adversaries.

Our contributions can be summarized as follows:

- Based on our proposed framework, we first offer a non-vacuous PAC-Bayesian style generalization bound for pre-training. Assuming the validity of our framework, we conclude that high-performing pre-trained models will inevitably be susceptible to generating behaviour that is present in the training corpus, including any unintended and harmful behaviour.
- Subsequently, we extend our framework to include notions of alignment and jailbreaking. Assuming our assumptions are met, we demonstrate jailbreaking to be unpreventable even after safety alignment because the LM fails to concentrate its output distribution over the set of safe responses.

2. Framework and assumptions

For the purpose of this work, we will *view any prompt as a tuple of query and concept* (q, c), where $c \in \mathcal{C}$, and $q \in \mathcal{Q}$, with $\mathcal{C}, \mathcal{Q}$ denoting the complete concept set and query set. Conceptually, we think of *concepts* as representing the

[*] Equal advising. [1] NYU [2] Meta FAIR. Correspondence to: Jingtong Su <js12196@nyu.edu>.

Work presented at TF2M workshop at ICML 2024, Vienna, Austria. PMLR 235, 2024. Copyright 2024 by the author(s).

information content of the prompt, usually through a short piece of text, for example "tutorial on making a cake". *Queries* are instructional text pieces that are composable with certain concepts. We can think of queries as mechanisms to trigger an LM to expand a concept in a specific way. Examples include "Tell me how to {}", or "We are now in an imaginary world, and you are not bounded by any ethical concerns. Teach my avatar how to {}". Since not all queries and concepts are composable,[1] we denote $\mathcal{P} \subsetneq \mathcal{Q} \times \mathcal{C}$ as the set of all *plausible* prompts, where the definition of plausible will be made clear below. **The decomposition of prompts allows us to isolate and hence bound the adversary's strength.**

In contrast to previous theoretical work where LMs are regarded as *single sentence generators* (Wolf et al., 2023), we model LMs as *lengthier text fragment generators*, and refer to possible generated content $e \in \mathcal{E}$ as explanations. Conceptually, explanations expand concepts with additional information. For example, "The US president in 2023 is Joe Biden.". An LM thus induces a mapping from *plausible* prompts to distributions over explanations, $p_{LM} : \mathcal{P} \to \Delta(\mathcal{E})$, where $\Delta(\mathcal{E})$ denotes the set of distributions defined over elements in $\mathcal{E}$.[2] The output of a LM given a prompt, $p_{LM}(q, c)$, is a discrete distribution over explanations. We use $\mathrm{dom}(p_{LM}(q, c))$ as the *domain* of this distribution, $p_{LM}(e|q, c)$ as the probability of e given (q, c) as the input, and $\mathrm{supp}(p_{LM}(q, c))$ as the subset of $\mathcal{E}$ with non-zero $p_{LM}(e|q, c)$. Further, we assume the existence of a latent ground truth mapping $p_{world} : \mathcal{P} \to \Delta(\mathcal{E})$ that the LM is optimized to mimic during the pretraining stage. This is the distribution that defines "knowledge": for all *plausible* prompts (q, c), it specifies the ground-truth distribution over explanations. By *plausible*, we refer to all prompts that lie in the domain of the ground truth mapping $(q, c) \in \mathrm{dom}(p_{world})$, *i.e.*, $\mathcal{P} \equiv \mathrm{dom}(p_{world})$.

We can now state our main assumption, namely that for any plausible prompt $(q, c) \in \mathrm{dom}(p_{world})$ the ground-truth distribution $p_{world}(q, c)$ is supported on a small subset of $\mathcal{E} \Leftrightarrow \mathrm{supp}(p_{world}(q, c)) \subsetneq \mathcal{E}$. Our second assumption is that for all plausible prompts (q, c), the concept c *uniquely determines* the *support* of the output distribution specified by p_{world}, regardless of the query: $\mathrm{supp}(p_{world}(q, c)) = \mathrm{supp}(p_{world}(q^*, c))$, $\forall$ plausible (q, c) and, (q^*, c) . The query changes the ground-truth distribution without affecting its support. An illustration is depicted in Figure 1 (see Appendix). To be more precise:

Assumption 2.1. *(Concepts uniquely determine the explanation for plausible prompts)*
For all plausible prompts $(q, c) \in \mathrm{dom}(p_{world})$,

$$\textit{i)}\ p_{world} : \mathcal{P} \to \Delta(\mathrm{supp}(p_{world}(q, c))$$

$$\textit{where}\ \mathrm{supp}(p_{world}(q, c)) \subsetneq \mathcal{E}\ \textit{s.t.}\ |\mathrm{supp}(p_{world}(q, c))| \ll |\mathcal{E}|;\ \textit{and}$$

$$\textit{ii)}\ \mathrm{supp}(p_{world}(q, c)) = \mathrm{supp}(p_{world}(q^*, c)),\ \forall (q, c), (q^*, c)\ \textit{plausible}.$$

This assumption is natural since it essentially tells us that knowledge is specified by the corresponding concept alone, irrespective of what query is used to extract it. In other words, given a concept c, if a query q manages to change $\mathrm{supp}(p_{world}(q, c))$, we argue that the query should be deconstructed and partially absorbed by c to accurately reflect the knowledge mirrored by the support.

Lastly, we make the assumption on the existence of an underlying generative distribution over prompts, denoted as $(q, c) \sim D_{\mathcal{P}}$. This distribution serves as the principle governing the creation of our pretraining corpus. It is important to note that $\mathrm{supp}(D_{\mathcal{P}}) \subsetneq \mathrm{dom}(p_{world})$. For example, take the prompt (q', c')=`"Who is James Bond $λ⋆#!48811"`; even though this prompt never appears in any text corpus across the internet, $(q', c') \notin \mathrm{supp}(D_{\mathcal{P}})$, we, as humans, can make sense of it: $(q', c') \in \mathrm{dom}(p_{world})$. Later proofs in this paper assume LMs generate semantically reasonable explanations for such unseen plausible prompts, since in reality LMs are claimed to generalize well on huge, out-of-distribution datasets (Srivastava et al., 2022). This is made explicit in Section 4, within Assumption 4.1.

Finally, the following definitions pertain to our notion of harmfulness. More specifically, we understand harmful behaviour abstractly as any unintended behaviour. For this, we assume that any explanation e can be denoted as **either harmful or not harmful (safe)**. A concept c is regarded as harmful if and only if the world generates harmful explanations with probability higher than a certain threshold with direct prompts.

Definition 2.1. *(Notions of Harmfulness)*

- ***(Direct Queries and Direct Prompts)*** *We refer to a prompt as direct if it stems from* $D_{\mathcal{P}}$, *i.e.*, $(q, c) \in \mathrm{supp}(D_{\mathcal{P}})$. *The query of a direct prompt is called a direct query.*
- ***(Harmful Concepts and Harmful Set)*** *Given a concept* c, *the associated harmful set of explanations is denoted as* $E_h(c) := \{e | e \in \mathrm{supp}(p_{world}(\cdot, c)) \wedge$ *e is harmful*$\}$. *In accordance with Assumption 2.1, with a threshold* η, *a concept* c *is harmful if* $\forall q$ *s.t.* $(q, c) \in \mathrm{dom}(p_{world})$, $\sum_{e: e \in E_h(c)} p_{world}(e|q, c) \geq 1 - \eta$. *We refer to the set of all possible harmful concepts as* $\mathcal{C}_h \subsetneq \mathcal{C}$.
- ***(Safe Set)*** $\forall c \in \mathcal{C}_h$, *there exists a corresponding* ***safe set*** $E_s(c) \subsetneq \mathcal{E}$ *that we wish* $p_{LM}(q, c)$ *to be concentrated on. It includes safe explanations existing in*

[1] For example, "Who is a tutorial on making a cake." is unreasonable.

[2] For real-world LMs, with different decoding hyperparameters *e.g.,* the temperature T, top-p and top-k sampling parameters, the induced distribution with the same set of parameters could be different. Our discussion holds for a pre-fixed set of hyperparameters throughout this paper.

$\mathrm{supp}(p_{world}(\cdot, c))$, *and explanations designed by humans,* e.g., *with the template beginning with "Sorry."*

- ***(Semantically meaningful)*** *We call explanations in $E_h(c) \cup E_s(c)$ as semantically meaningful for the (q, c) prompt.*
- ***(Mixture decomposition of $D_{\mathcal{P}}$)*** *With these notions, we can decompose $D_{\mathcal{P}} = \alpha D_{\mathcal{P}_h} + (1-\alpha) D_{\mathcal{P}_s}$ (where $\mathrm{supp}(D_{\mathcal{P}_h})$ includes all direct prompts with a harmful concept, and $\mathrm{supp}(D_{\mathcal{P}_s})$ includes the complement) as a mixture over direct prompts with a harmful concept and the non-harmful counterpart.*

3. PAC-Bayesian bound for pre-training LLMs on harmful data

Given a learning algorithm that leads to a *posterior distribution* over a set of models, PAC-Bayesian theory (McAllester, 1998) provide bounds on the generalization gap, *i.e.,* the difference between the model's empirical loss and the population loss. We now present the first result of our analysis: a non-vacuous PAC-Bayesian bound for pretraining LMs which implies that a well-trained LM ought to exhibit harmful behaviour even when simply prompted with direct queries if it was presented with harmful behavior during training.

We denote by $S = \{(q_i, c_i)\}_{i=1}^n$ a set of prompts generated *i.i.d.* under $D_{\mathcal{P}}$, $S \sim D_{\mathcal{P}}^n$. These prompts together with sampled explanations form our pretraining corpus. We use π, ρ as the prior and posterior distribution over LMs before and after the pretraining process, defined over $\mathbb{LM}$, the set of language models. Given a prompt (q, c), we measure the generalization capability of a LM by quantifying the Total Variation (TV) loss between the induced distribution $p_{LM}(q, c)$ and the ground-truth distribution $p_{world}(q, c)$.[3] For real-world LMs, pretraining involves optimizing the cross-entropy loss on the training corpus, which is equivalent to minimizing $\mathrm{KL}[p_{world}(q, c) || p_{LM}(q, c)]$ under our framework. With Pinsker's Inequality, optimizing the KL-divergence term is equivalent to optimizing an upper bound on TV; thus we expect empirical TV loss be small.

Definition 3.1. *(TV empirical loss and population loss)*

$$\ell_{\mathrm{TV}}(p_{LM}, (q, c)) := \mathrm{TV}(p_{world}(q, c), p_{LM}(q, c)).$$

Given an LM and a set of data S, the empirical loss $\hat{R}_S(p_{LM})$ and population loss $R(p_{LM})$ are defined as

$$\hat{R}_S(p_{LM}) := \frac{1}{n} \sum_{i=1}^{n} \ell_{\mathrm{TV}}(p_{LM}, (q_i, c_i));$$
$$R(p_{LM}) := \mathbb{E}_{S \sim D_{\mathcal{P}}^n} \left[\hat{R}_S(p_{LM})\right] = \mathbb{E}_{(q,c)\sim D_{\mathcal{P}}} \left[\ell_{\mathrm{TV}}(p_{LM}, (q, c))\right].$$

We state our PAC-Bayesian bound as follows. The detailed proof can be found in Appendix C.1.

Theorem 1. *(PAC-Bayesian Generalization Bound for Language Models.) With α as in Definition 2.1, consider a set of language models $\mathbb{LM}$, with prior distribution π over $\mathbb{LM}$.*

Given any $\delta \in (0, 1)$, for any probability measure ρ over $\mathbb{LM}$ such that ρ, π share the same support, the following holds with probability at least $1 - \delta$ over the random draw of S:

$$\mathbb{E}_{LM\sim\rho}[R(p_{LM}) - \hat{R}_S(p_{LM})] \leq \sqrt{\frac{[\mathrm{KL}[\rho||\pi] + \log\frac{1}{\delta}]}{2n}} := \varrho;$$

$$\mathbb{E}_{LM\sim\rho}[\mathbb{E}_{(q,c)\sim D_{\mathcal{P}_h}} \ell_{\mathrm{TV}}(p_{LM}, (q, c))] \leq \frac{1}{\alpha}\left[\mathbb{E}_{LM\sim\rho}\hat{R}_S(p_{LM}) + \varrho\right].$$

In Appendix C.2 we give a theoretical estimation of ϱ, to illustrate the bound we derive is non-vacuous, *i.e.,* less than 1. Theorem 1 tells us that, as long as pretraining successfully reduces the loss on the training corpus ($\hat{R}_S(p_{LM}) \downarrow$), in expectation the language model will mimic the world well (small ℓ_{TV} difference) on a given direct prompt sampled from $D_{\mathcal{P}}$. Furthermore, if α is not too small, then this statement holds on a direct prompt whose concept is harmful.

4. A statistical perspective on jailbreaking after alignment

In this section, we will present the second result that, given our assumptions hold, we prove the *existence of ways for an adversary to jailbreak an LM even after the preference alignment process*. Going forward, we need to extend our framework to integrate alignment and jailbreaking.

After an LM is pretrained, it typically will undergo finetuning on a dataset containing preferred behaviour. In what follows, we will assume that this alignment process does not change the model performance in the sense that the LM will still produce semantically meaningful explanations (Definition 2.1). It would not, for example, default to answering any request with the same response.

Assumption 4.1. *(LM outputs semantically meaningful explanations) For any harmful concept c, and all plausible prompts $(q, c) \in \mathrm{dom}(p_{world})$,*

$$\exists\, |E_n(c)| \ll |E_h(c)| + |E_s(c)| \text{ s.t.}$$
$$O(1) \ll |\mathrm{dom}(p_{LM}(q, c))| = |E_h(c) \cup E_s(c) \cup E_n(c)|.$$

In other words, we assume the LM's output distribution is accurately supported on $E_h(c) \cup E_s(c)$, in the sense that the size of "residual" $E_n(c)$ is relatively small compared to these semantically meaningful explanations. We define $n(c) = |E_n(c)| + |E_s(c)| + |E_h(c)|$. We omit the (c) annotations when clear from the context. We discuss the validity of this assumption in Appendix C.3.

To bound the likelihood of jailbreaking we first need to specify how the output of a LM interacts with its support. Assuming a fixed order of explanations in $\mathrm{dom}(p_{LM}(q, c))$, and slight abuse of notation, we can use $p_{LM}(q, c)$ to denote an $n(c)$-dimensional vector on $\Delta^{n(c)-1}$, the probability

[3] We regard both distributions as defined over the entire $\mathcal{E}$ since we do not restrict the output distribution of LM in this section.

simplex with $n(c)$ elements, where each entry represents the probability of a single explanation. We call this simplex the **output simplex** related to a given concept c. Next, we can induce a distribution on this simplex given a posterior distribution γ over the set of language models $\mathbb{LM}$, as follows.

Definition 4.1. *(**Induced Distribution on Simplex,** γ_c) Under the assumption that the LM outputs semantically meaningful explanations (Assumption 4.1), with a fixed prompt (q, c) and a posterior distribution γ over $\mathbb{LM}$, the corresponding induced distribution: $p_{LM}(q, c)$ where $LM \sim \gamma$ is supported over a subset of the output simplex Δ^{n-1}. This distribution is denoted as $\gamma_{(q,c)}$, or γ_c when the reference to q is clear from context.*

Next, we will separate the output simplex into a harmful and safety zone. This definition is motivated by the observation that typically an adversary is deemed successful if it can extract even a single harmful explanation for a given concept. This translates into a division of the output simplex, under Assumption 4.1, as follows.

Definition 4.2. *(**Harmful Zone and Safety Zone**) For a given harmful concept c and a fixed LM, the output simplex is divided into a **safety zone and a harmful zone**, $\mathcal{H}_s$ and $\mathcal{H}_h$, where a* threshold *$p \in [0, 1]$ is used to quantify the distinction: $p_{LM}(q, c) \in \mathcal{H}_h$ iff $\sum_{e:e \in E_h(c)} p_{LM}(e|q, c) \geq p$, and otherwise $p_{LM}(q, c) \in \mathcal{H}_s$.*

Before we introduce jailbreaking, the reader might wonder why we did not define alignment more clearly. This is because under the PAC framework, preference alignment is nothing but a transformation from ρ to some γ posterior defined over $\mathbb{LM}$. Given this inability on fine-grained characterization of alignment, we instead provide the *goal of it* as follows. With the above notion, given a prompt (q, c) where c is harmful, its goal is to push the induced distribution γ_c into the safety zone $\mathcal{H}_s$. Ideally, $\mathrm{supp}(\gamma_c) \subset \mathcal{H}_s \Leftrightarrow$ with probability 1, the resulting LM is safe when encountering (q, c). We are ready to introduce necessary concepts related to jailbreaking.

Definition 4.3. *(**Jailbreaking**) Given a harmful concept c and a query q', the prompt (q', c) **jailbreaks** the LM iff $p_{LM}(q', c) \in \mathcal{H}_h$. We call such a prompt (q', c) and query q' a jailbreaking prompt and jailbreaking query, respectively.*

To theoretically prove the jailbreaking effect, we need to restrict the adversary's ability. To achieve this goal, we borrow insights from adversarial attacks, to assume that the adversary has bounded manipulating capability on the output simplex when searching over the query set:

Assumption 4.2. *(ϵ-bounded adversary) Given an LM, a harmful concept c and an associated direct prompt (q, c), we assume the adversary can find a set of queries $\mathcal{Q}'$, such that the output is moved **at most** ϵ on the simplex towards $\mathcal{H}_h$ from $p_{LM}(q, c)$:*

$$\sup_{q' \in \mathcal{Q}'} d(p_{LM}(q, c), p_{LM}(q', c)) = \epsilon.$$

Here d is a distance measure between two discrete distributions. d can be a typical ℓ_p measure with $p \geq 1$, or the Total Variation / Jensen-Shannon Divergence. We call $q' \in \mathcal{Q}'$ an ϵ-bounded query.

A conceptual illustration of our framework is depicted in Figure 2 (see Appendix). Before arriving at our Theorem, we give the final definition of ϵ-expansion.

Definition 4.4. *(ϵ-expansion) Given a set $A \subset \Delta^{n-1}$ and a distance measure d, the ϵ-expansion set $A(\epsilon, d)$ is defined as*

$$A(\epsilon, d) := \{t | t \in \Delta^{n-1} \wedge \exists y \in A \ s.t. \ ||y - t||_d \leq \epsilon\}.$$

We are ready to present the following theorem, which states that as long as the induced posterior γ_c is not concentrated in an extremely safe area, then with high probability the model can be jailbroken. The proof is in Appendix C.4.

Theorem 2. *(Jailbreak is unavoidable) Assume that an LMs output semantically meaningful explanations (Assumption 4.1). Given any γ posterior distribution over $\mathbb{LM}$, choose a harmful concept c with a direct prompt (q, c) and a threshold p (Definition 2.1), to define the corresponding induced distribution γ_c (Definition 4.1) and division over output simplex (Definition 4.2). An ϵ-bounded adversary (Assumption 4.2) can find a jailbreaking prompt (Definition 4.3) with probability at least*

$$1 - \gamma_s \times (1 - \Phi(a_\epsilon)),$$

- *by using either the direct prompt, such that $p_{LM}(q, c) \in \mathcal{H}_h$; or*
- *by finding an ϵ-bounded query q', such that $p_{LM}(q', c) \in \mathcal{H}_h$.*

Here, $\Phi(\cdot)$ is the standard Gaussian cdf, $\gamma_s := \max_{x \in \mathcal{H}_s - \mathcal{H}_h(\epsilon, d)} \frac{\gamma_c(x)}{U(x)}$, with $U(x)$ the uniform distribution over Δ^{n-1}, and $a_\epsilon := a + \sqrt{n-1}\epsilon$, where a writes analytically as $a \asymp \frac{|E_h(c)| - 1 - (n-1)p}{\sqrt{(n-1)p(1-p)}}$.

Trivially, the chances of an adversary to find a jailbreaking prompt increase for stronger adversaries (ϵ ↑). In the real world, this could relate to how much compute budget we allow to alter a query for a specific harmful concept. Furthermore, the chances of an adversary to find a jailbreaking prompt increase when the ratio of the sizes of the harmful explanation set to the safe explanation set is larger $\frac{|E_h(c)|}{|E_s(c)|} \uparrow$. This is because their ratio will determine the size of the harmful zone which in turn will cause $\Phi(a_\epsilon) \to 1$. In real world settings, for any harmful concept, the training corpus naturally contains a large harmful set due to the number of possible responses. Realistically, its size can not be countered by any manually-constructed safe set. **Hence achieving alignment is hard**: Recall that the goal of alignment is to respond with only safe explanations with high probability. However, we just learned that to increase that probability, we need to have a small harmful-to-safety set ratio which we discussed is not realistic. Consequently, the safety zone is going to be small.

References

Baptiste Roziere, Jonas Gehring, Fabian Gloeckle, Sten Sootla, Itai Gat, Xiaoqing Ellen Tan, Yossi Adi, Jingyu Liu, Tal Remez, Jérémy Rapin, et al. Code llama: Open foundation models for code. *arXiv preprint arXiv:2308.12950*, 2023. 1

Karan Singhal, Shekoofeh Azizi, Tao Tu, S Sara Mahdavi, Jason Wei, Hyung Won Chung, Nathan Scales, Ajay Tanwani, Heather Cole-Lewis, Stephen Pfohl, et al. Large language models encode clinical knowledge. *Nature*, 620 (7972):172–180, 2023. 1

Kaiyu Yang, Aidan Swope, Alex Gu, Rahul Chalamala, Peiyang Song, Shixing Yu, Saad Godil, Ryan J Prenger, and Animashree Anandkumar. Leandojo: Theorem proving with retrieval-augmented language models. *Advances in Neural Information Processing Systems*, 36, 2024. 1

Emily M Bender, Timnit Gebru, Angelina McMillan-Major, and Shmargaret Shmitchell. On the dangers of stochastic parrots: Can language models be too big? In *Proceedings of the 2021 ACM conference on fairness, accountability, and transparency*, pages 610–623, 2021. 1

Julian Hazell. Large language models can be used to effectively scale spear phishing campaigns. *arXiv preprint arXiv:2305.06972*, 2023. 1

Yi Liu, Gelei Deng, Zhengzi Xu, Yuekang Li, Yaowen Zheng, Ying Zhang, Lida Zhao, Tianwei Zhang, and Yang Liu. Jailbreaking chatgpt via prompt engineering: An empirical study. *arXiv preprint arXiv:2305.13860*, 2023a. 1, 11

OpenAI. Chatgpt. https://openai.com/index/chatgpt/, 2022. 1

DAN. Do anything now prompt. https://github.com/0xk1h0/ChatGPT_DAN, 2023. 1

Reddit. Chatgpt grandma exploit. https://www.reddit.com/r/ChatGPT/comments/12sn0kk/grandma_exploit/, 2023. 1

Yotam Wolf, Noam Wies, Yoav Levine, and Amnon Shashua. Fundamental limitations of alignment in large language models. *arXiv preprint arXiv:2304.11082*, 2023. 2, 14

Aarohi Srivastava, Abhinav Rastogi, Abhishek Rao, Abu Awal Md Shoeb, Abubakar Abid, Adam Fisch, Adam R Brown, Adam Santoro, Aditya Gupta, Adrià Garriga-Alonso, et al. Beyond the imitation game: Quantifying and extrapolating the capabilities of language models. *arXiv preprint arXiv:2206.04615*, 2022. 2

David A McAllester. Some pac-bayesian theorems. In *Proceedings of the eleventh annual conference on Computational learning theory*, pages 230–234, 1998. 3

Taylor Shin, Yasaman Razeghi, Robert L Logan IV, Eric Wallace, and Sameer Singh. Autoprompt: Eliciting knowledge from language models with automatically generated prompts. *arXiv preprint arXiv:2010.15980*, 2020. 10, 11

Erik Jones, Anca Dragan, Aditi Raghunathan, and Jacob Steinhardt. Automatically auditing large language models via discrete optimization. *arXiv preprint arXiv:2303.04381*, 2023. 10

Chuan Guo, Alexandre Sablayrolles, Hervé Jégou, and Douwe Kiela. Gradient-based adversarial attacks against text transformers. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 5747–5757, 2021. 10

Fábio Perez and Ian Ribeiro. Ignore previous prompt: Attack techniques for language models. *arXiv preprint arXiv:2211.09527*, 2022. 11

Xinyue Shen, Zeyuan Chen, Michael Backes, Yun Shen, and Yang Zhang. " do anything now": Characterizing and evaluating in-the-wild jailbreak prompts on large language models. *arXiv preprint arXiv:2308.03825*, 2023. 11

Andy Zou, Zifan Wang, J Zico Kolter, and Matt Fredrikson. Universal and transferable adversarial attacks on aligned language models. *arXiv preprint arXiv:2307.15043*, 2023. 11, 12, 13

Alexander Wei, Nika Haghtalab, and Jacob Steinhardt. Jailbroken: How does llm safety training fail? In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023a. 11

Xiaogeng Liu, Nan Xu, Muhao Chen, and Chaowei Xiao. Autodan: Generating stealthy jailbreak prompts on aligned large language models. *arXiv preprint arXiv:2310.04451*, 2023b. 11

Raz Lapid, Ron Langberg, and Moshe Sipper. Open sesame! universal black box jailbreaking of large language models. *arXiv preprint arXiv:2309.01446*, 2023. 11

Patrick Chao, Alexander Robey, Edgar Dobriban, Hamed Hassani, George J Pappas, and Eric Wong. Jailbreaking black box large language models in twenty queries. *arXiv preprint arXiv:2310.08419*, 2023. 11

Sicheng Zhu, Ruiyi Zhang, Bang An, Gang Wu, Joe Barrow, Zichao Wang, Furong Huang, Ani Nenkova, and

Tong Sun. Autodan: Automatic and interpretable adversarial attacks on large language models. *arXiv preprint arXiv:2310.15140*, 2023. 11

Xiaoxia Li, Siyuan Liang, Jiyi Zhang, Han Fang, Aishan Liu, and Ee-Chien Chang. Semantic mirror jailbreak: Genetic algorithm based jailbreak prompts against open-source llms. *arXiv preprint arXiv:2402.14872*, 2024a. 11

Gelei Deng, Yi Liu, Yuekang Li, Kailong Wang, Ying Zhang, Zefeng Li, Haoyu Wang, Tianwei Zhang, and Yang Liu. Jailbreaker: Automated jailbreak across multiple large language model chatbots. *arXiv preprint arXiv:2307.08715*, 2023a. 11

Jiahao Yu, Xingwei Lin, and Xinyu Xing. Gptfuzzer: Red teaming large language models with auto-generated jailbreak prompts. *arXiv preprint arXiv:2309.10253*, 2023. 11

Peng Ding, Jun Kuang, Dan Ma, Xuezhi Cao, Yunsen Xian, Jiajun Chen, and Shujian Huang. A wolf in sheep's clothing: Generalized nested jailbreak prompts can fool large language models easily. *arXiv preprint arXiv:2311.08268*, 2023. 11

Anay Mehrotra, Manolis Zampetakis, Paul Kassianik, Blaine Nelson, Hyrum Anderson, Yaron Singer, and Amin Karbasi. Tree of attacks: Jailbreaking black-box llms automatically. *arXiv preprint arXiv:2312.02119*, 2023. 11

Shunyu Yao, Dian Yu, Jeffrey Zhao, Izhak Shafran, Tom Griffiths, Yuan Cao, and Karthik Narasimhan. Tree of thoughts: Deliberate problem solving with large language models. *Advances in Neural Information Processing Systems*, 36, 2024. 11

Xuan Li, Zhanke Zhou, Jianing Zhu, Jiangchao Yao, Tongliang Liu, and Bo Han. Deepinception: Hypnotize large language model to be jailbreaker. *arXiv preprint arXiv:2311.03191*, 2023a. 11

Xirui Li, Ruochen Wang, Minhao Cheng, Tianyi Zhou, and Cho-Jui Hsieh. Drattack: Prompt decomposition and reconstruction makes powerful llm jailbreakers. *arXiv preprint arXiv:2402.16914*, 2024b. 11

Zhenhua Wang, Wei Xie, Baosheng Wang, Enze Wang, Zhiwen Gui, Shuoyoucheng Ma, and Kai Chen. Foot in the door: Understanding large language model jailbreaking via cognitive psychology. *arXiv preprint arXiv:2402.15690*, 2024a. 11

Anselm Paulus, Arman Zharmagambetov, Chuan Guo, Brandon Amos, and Yuandong Tian. Advprompter: Fast adaptive adversarial prompting for llms. *arXiv preprint arXiv:2404.16873*, 2024. 11

Yangsibo Huang, Samyak Gupta, Mengzhou Xia, Kai Li, and Danqi Chen. Catastrophic jailbreak of open-source llms via exploiting generation. *arXiv preprint arXiv:2310.06987*, 2023. 11, 16

Hangfan Zhang, Zhimeng Guo, Huaisheng Zhu, Bochuan Cao, Lu Lin, Jinyuan Jia, Jinghui Chen, and Dinghao Wu. On the safety of open-sourced large language models: Does alignment really prevent them from being misused? *arXiv preprint arXiv:2310.01581*, 2023a. 11

Xuandong Zhao, Xianjun Yang, Tianyu Pang, Chao Du, Lei Li, Yu-Xiang Wang, and William Yang Wang. Weak-to-strong jailbreaking on large language models. *arXiv preprint arXiv:2401.17256*, 2024. 11

Xianjun Yang, Xiao Wang, Qi Zhang, Linda Petzold, William Yang Wang, Xun Zhao, and Dahua Lin. Shadow alignment: The ease of subverting safely-aligned language models. *arXiv preprint arXiv:2310.02949*, 2023. 12

Qiusi Zhan, Richard Fang, Rohan Bindu, Akul Gupta, Tatsunori Hashimoto, and Daniel Kang. Removing rlhf protections in gpt-4 via fine-tuning. *arXiv preprint arXiv:2311.05553*, 2023. 12

Xiangyu Qi, Yi Zeng, Tinghao Xie, Pin-Yu Chen, Ruoxi Jia, Prateek Mittal, and Peter Henderson. Fine-tuning aligned language models compromises safety, even when users do not intend to! *arXiv preprint arXiv:2310.03693*, 2023a. 12

Zheng-Xin Yong, Cristina Menghini, and Stephen H Bach. Low-resource languages jailbreak gpt-4. *arXiv preprint arXiv:2310.02446*, 2023. 12

Yue Deng, Wenxuan Zhang, Sinno Jialin Pan, and Lidong Bing. Multilingual jailbreak challenges in large language models. *arXiv preprint arXiv:2310.06474*, 2023b. 12

Youliang Yuan, Wenxiang Jiao, Wenxuan Wang, Jen-tse Huang, Pinjia He, Shuming Shi, and Zhaopeng Tu. Gpt-4 is too smart to be safe: Stealthy chat with llms via cipher. *arXiv preprint arXiv:2308.06463*, 2023. 12

Erfan Shayegani, Yue Dong, and Nael Abu-Ghazaleh. Jailbreak in pieces: Compositional adversarial attacks on multi-modal language models. *arXiv preprint arXiv:2307.14539*, 2023. 12

Xiangyu Qi, Kaixuan Huang, Ashwinee Panda, Mengdi Wang, and Prateek Mittal. Visual adversarial examples jailbreak large language models. *arXiv preprint arXiv:2306.13213*, 2023b. 12

Nicholas Carlini, Milad Nasr, Christopher A Choquette-Choo, Matthew Jagielski, Irena Gao, Pang Wei Koh, Daphne Ippolito, Florian Tramèr, and Ludwig Schmidt. Are aligned neural networks adversarially aligned? In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023. 12

Natalie Maus, Patrick Chao, Eric Wong, and Jacob R Gardner. Black box adversarial prompting for foundation models. In *The Second Workshop on New Frontiers in Adversarial Machine Learning*, 2023. 12

Zeming Wei, Yifei Wang, and Yisen Wang. Jailbreak and guard aligned language models with only few in-context demonstrations. *arXiv preprint arXiv:2310.06387*, 2023b. 12

Jiongxiao Wang, Zichen Liu, Keun Hee Park, Muhao Chen, and Chaowei Xiao. Adversarial demonstration attacks on large language models. *arXiv preprint arXiv:2305.14950*, 2023a. 12

Wei-Lin Chiang, Zhuohan Li, Zi Lin, Ying Sheng, Zhanghao Wu, Hao Zhang, Lianmin Zheng, Siyuan Zhuang, Yonghao Zhuang, Joseph E. Gonzalez, Ion Stoica, and Eric P. Xing. Vicuna: An open-source chatbot impressing gpt-4 with 90%* chatgpt quality, March 2023. URL https://lmsys.org/blog/2023-03-30-vicuna/. 12

Cem Anil, Esin Durmus, Mrinank Sharma, Joe Benton, Sandipan Kundu, Joshua Batson, Nina Rimsky, Meg Tong, Jesse Mu, Daniel Ford, et al. Many-shot jailbreaking, 2024. 12

Daniel Kang, Xuechen Li, Ion Stoica, Carlos Guestrin, Matei Zaharia, and Tatsunori Hashimoto. Exploiting programmatic behavior of llms: Dual-use through standard security attacks. *arXiv preprint arXiv:2302.05733*, 2023. 12

Rusheb Shah, Soroush Pour, Arush Tagade, Stephen Casper, Javier Rando, et al. Scalable and transferable black-box jailbreaks for language models via persona modulation. *arXiv preprint arXiv:2311.03348*, 2023. 12

Yi Zeng, Hongpeng Lin, Jingwen Zhang, Diyi Yang, Ruoxi Jia, and Weiyan Shi. How johnny can persuade llms to jailbreak them: Rethinking persuasion to challenge ai safety by humanizing llms. *arXiv preprint arXiv:2401.06373*, 2024a. 12

Xingang Guo, Fangxu Yu, Huan Zhang, Lianhui Qin, and Bin Hu. Cold-attack: Jailbreaking llms with stealthiness and controllability. *arXiv preprint arXiv:2402.08679*, 2024. 12

Lianhui Qin, Sean Welleck, Daniel Khashabi, and Yejin Choi. Cold decoding: Energy-based constrained text generation with langevin dynamics. *Advances in Neural Information Processing Systems*, 35:9538–9551, 2022. 12

Somnath Banerjee, Sayan Layek, Rima Hazra, and Animesh Mukherjee. How (un) ethical are instruction-centric responses of llms? unveiling the vulnerabilities of safety guardrails to harmful queries. *arXiv preprint arXiv:2402.15302*, 2024. 12

Neal Mangaokar, Ashish Hooda, Jihye Choi, Shreyas Chandrashekaran, Kassem Fawaz, Somesh Jha, and Atul Prakash. Prp: Propagating universal perturbations to attack large language model guard-rails. *arXiv preprint arXiv:2402.15911*, 2024. 12

Huijie Lv, Xiao Wang, Yuansen Zhang, Caishuang Huang, Shihan Dou, Junjie Ye, Tao Gui, Qi Zhang, and Xuanjing Huang. Codechameleon: Personalized encryption framework for jailbreaking large language models. *arXiv preprint arXiv:2402.16717*, 2024. 12

Vinu Sankar Sadasivan, Shoumik Saha, Gaurang Sriramanan, Priyatham Kattakinda, Atoosa Chegini, and Soheil Feizi. Fast adversarial attacks on language models in one gpu minute. *arXiv preprint arXiv:2402.15570*, 2024. 12

Jonas Geiping, Alex Stein, Manli Shu, Khalid Saifullah, Yuxin Wen, and Tom Goldstein. Coercing llms to do and reveal (almost) anything. *arXiv preprint arXiv:2402.14020*, 2024. 12

Qibing Ren, Chang Gao, Jing Shao, Junchi Yan, Xin Tan, Wai Lam, and Lizhuang Ma. Exploring safety generalization challenges of large language models via code, 2024. 12

Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks. In *International Conference on Learning Representations*, 2018. 12

Martin Abadi, Andy Chu, Ian Goodfellow, H Brendan McMahan, Ilya Mironov, Kunal Talwar, and Li Zhang. Deep learning with differential privacy. In *Proceedings of the 2016 ACM SIGSAC conference on computer and communications security*, pages 308–318, 2016. 12

Daniel M Ziegler, Nisan Stiennon, Jeffrey Wu, Tom B Brown, Alec Radford, Dario Amodei, Paul Christiano, and Geoffrey Irving. Fine-tuning language models from human preferences. *arXiv preprint arXiv:1909.08593*, 2019. 12

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. Training language models to follow instructions with human feedback. *Advances in Neural Information Processing Systems*, 35:27730–27744, 2022. 12

Yuntao Bai, Andy Jones, Kamal Ndousse, Amanda Askell, Anna Chen, Nova DasSarma, Dawn Drain, Stanislav Fort, Deep Ganguli, Tom Henighan, et al. Training a helpful and harmless assistant with reinforcement learning from human feedback. *arXiv preprint arXiv:2204.05862*, 2022a. 12

Tomasz Korbak, Kejian Shi, Angelica Chen, Rasika Vinayak Bhalerao, Christopher Buckley, Jason Phang, Samuel R Bowman, and Ethan Perez. Pretraining language models with human preferences. In *International Conference on Machine Learning*, pages 17506–17533. PMLR, 2023. 12

Yuntao Bai, Saurav Kadavath, Sandipan Kundu, Amanda Askell, Jackson Kernion, Andy Jones, Anna Chen, Anna Goldie, Azalia Mirhoseini, Cameron McKinnon, et al. Constitutional ai: Harmlessness from ai feedback. *arXiv preprint arXiv:2212.08073*, 2022b. 12

Harrison Lee, Samrat Phatale, Hassan Mansoor, Kellie Lu, Thomas Mesnard, Colton Bishop, Victor Carbune, and Abhinav Rastogi. Rlaif: Scaling reinforcement learning from human feedback with ai feedback. *arXiv preprint arXiv:2309.00267*, 2023. 12

Deep Ganguli, Liane Lovitt, Jackson Kernion, Amanda Askell, Yuntao Bai, Saurav Kadavath, Ben Mann, Ethan Perez, Nicholas Schiefer, Kamal Ndousse, et al. Red teaming language models to reduce harms: Methods, scaling behaviors, and lessons learned. *arXiv preprint arXiv:2209.07858*, 2022. 12

Ethan Perez, Saffron Huang, Francis Song, Trevor Cai, Roman Ring, John Aslanides, Amelia Glaese, Nat McAleese, and Geoffrey Irving. Red teaming language models with language models. *arXiv preprint arXiv:2202.03286*, 2022. 12

Stephen Casper, Jason Lin, Joe Kwon, Gatlen Culp, and Dylan Hadfield-Menell. Explore, establish, exploit: Red teaming language models from scratch. *arXiv preprint arXiv:2306.09442*, 2023. 12

Zhang-Wei Hong, Idan Shenfeld, Tsun-Hsuan Wang, Yung-Sung Chuang, Aldo Pareja, James R Glass, Akash Srivastava, and Pulkit Agrawal. Curiosity-driven red-teaming for large language models. In *The Twelfth International Conference on Learning Representations*, 2023. 12

Mikayel Samvelyan, Sharath Chandra Raparthy, Andrei Lupu, Eric Hambro, Aram H Markosyan, Manish Bhatt, Yuning Mao, Minqi Jiang, Jack Parker-Holder, Jakob Foerster, et al. Rainbow teaming: Open-ended generation of diverse adversarial prompts. *arXiv preprint arXiv:2402.16822*, 2024. 12

Gabriel Alon and Michael Kamfonas. Detecting language model attacks with perplexity. *arXiv preprint arXiv:2308.14132*, 2023. 12

Guolin Ke, Qi Meng, Thomas Finley, Taifeng Wang, Wei Chen, Weidong Ma, Qiwei Ye, and Tie-Yan Liu. Lightgbm: A highly efficient gradient boosting decision tree. *Advances in neural information processing systems*, 30, 2017. 13

Neel Jain, Avi Schwarzschild, Yuxin Wen, Gowthami Somepalli, John Kirchenbauer, Ping-yeh Chiang, Micah Goldblum, Aniruddha Saha, Jonas Geiping, and Tom Goldstein. Baseline defenses for adversarial attacks against aligned language models. *arXiv preprint arXiv:2309.00614*, 2023. 13

Alexander Robey, Eric Wong, Hamed Hassani, and George J Pappas. Smoothllm: Defending large language models against jailbreaking attacks. *arXiv preprint arXiv:2310.03684*, 2023. 13

Bochuan Cao, Yuanpu Cao, Lu Lin, and Jinghui Chen. Defending against alignment-breaking attacks via robustly aligned llm. *arXiv preprint arXiv:2309.14348*, 2023. 13

Aounon Kumar, Chirag Agarwal, Suraj Srinivas, Soheil Feizi, and Hima Lakkaraju. Certifying llm safety against adversarial prompting. *arXiv preprint arXiv:2309.02705*, 2023. 13

Yueqi Xie, Jingwei Yi, Jiawei Shao, Justin Curl, Lingjuan Lyu, Qifeng Chen, Xing Xie, and Fangzhao Wu. Defending chatgpt against jailbreak attack via self-reminders. *Nature Machine Intelligence*, pages 1–11, 2023. 13

Chujie Zheng, Fan Yin, Hao Zhou, Fandong Meng, Jie Zhou, Kai-Wei Chang, Minlie Huang, and Nanyun Peng. Prompt-driven llm safeguarding via directed representation optimization. *arXiv preprint arXiv:2401.18018*, 2024. 13

Andy Zhou, Bo Li, and Haohan Wang. Robust prompt optimization for defending language models against jailbreaking attacks. *arXiv preprint arXiv:2401.17263*, 2024a. 13

Yichuan Mo, Yuji Wang, Zeming Wei, and Yisen Wang. Studious bob fight back against jailbreaking via prompt adversarial tuning. *arXiv preprint arXiv:2402.06255*, 2024. 13

Zhexin Zhang, Junxiao Yang, Pei Ke, and Minlie Huang. Defending large language models against jailbreaking attacks through goal prioritization. *arXiv preprint arXiv:2311.09096*, 2023b. 13

Yujun Zhou, Yufei Han, Haomin Zhuang, Taicheng Guo, Kehan Guo, Zhenwen Liang, Hongyan Bao, and Xiangliang Zhang. Defending jailbreak prompts via in-context adversarial game. *arXiv preprint arXiv:2402.13148*, 2024b. 13

Xiaotian Zou, Yongkang Chen, and Ke Li. Is the system message really important to jailbreaks in large language models? *arXiv preprint arXiv:2402.14857*, 2024. 13

Alec Helbling, Mansi Phute, Matthew Hull, and Duen Horng Chau. Llm self defense: By self examination, llms know they are being tricked. *arXiv preprint arXiv:2308.07308*, 2023. 13

Zezhong Wang, Fangkai Yang, Lu Wang, Pu Zhao, Hongru Wang, Liang Chen, Qingwei Lin, and Kam-Fai Wong. Self-guard: Empower the llm to safeguard itself. *arXiv preprint arXiv:2310.15851*, 2023b. 13

Yuhui Li, Fangyun Wei, Jinjing Zhao, Chao Zhang, and Hongyang Zhang. Rain: Your language models can align themselves without finetuning. *arXiv preprint arXiv:2309.07124*, 2023b. 13

Zhangchen Xu, Fengqing Jiang, Luyao Niu, Jinyuan Jia, Bill Yuchen Lin, and Radha Poovendran. Safedecoding: Defending against jailbreak attacks via safety-aware decoding. *arXiv preprint arXiv:2402.08983*, 2024. 13

Adib Hasan, Ileana Rugina, and Alex Wang. Pruning for protection: Increasing jailbreak resistance in aligned llms without fine-tuning. *arXiv preprint arXiv:2401.10862*, 2024. 13

Mingjie Sun, Zhuang Liu, Anna Bair, and J Zico Kolter. A simple and effective pruning approach for large language models. *arXiv preprint arXiv:2306.11695*, 2023. 13

Renjie Pi, Tianyang Han, Yueqi Xie, Rui Pan, Qing Lian, Hanze Dong, Jipeng Zhang, and Tong Zhang. Mllm-protector: Ensuring mllm's safety without hurting performance. *arXiv preprint arXiv:2401.02906*, 2024. 13

Yuqi Zhang, Liang Ding, Lefei Zhang, and Dacheng Tao. Intention analysis prompting makes large language models a good jailbreak defender. *arXiv preprint arXiv:2401.06561*, 2024. 13

Yihan Wang, Zhouxing Shi, Andrew Bai, and Cho-Jui Hsieh. Defending llms against jailbreaking attacks via backtranslation. *arXiv preprint arXiv:2402.16459*, 2024b. 13

Heegyu Kim, Sehyun Yuk, and Hyunsouk Cho. Break the breakout: Reinventing lm defense against jailbreak attacks with self-refinement. *arXiv preprint arXiv:2402.15180*, 2024. 13

Yifan Zeng, Yiran Wu, Xiao Zhang, Huazheng Wang, and Qingyun Wu. Autodefense: Multi-agent llm defense against jailbreak attacks, 2024b. 13

Xiaomeng Hu, Pin-Yu Chen, and Tsung-Yi Ho. Gradient cuff: Detecting jailbreak attacks on large language models by exploring refusal loss landscapes. *arXiv preprint arXiv:2403.00867*, 2024. 13

Jiabao Ji, Bairu Hou, Alexander Robey, George J Pappas, Hamed Hassani, Yang Zhang, Eric Wong, and Shiyu Chang. Defending large language models against jailbreak attacks via semantic smoothing. *arXiv preprint arXiv:2402.16192*, 2024. 13

Hakan Inan, Kartikeya Upasani, Jianfeng Chi, Rashi Rungta, Krithika Iyer, Yuning Mao, Michael Tontchev, Qing Hu, Brian Fuller, Davide Testuggine, et al. Llama guard: Llm-based input-output safeguard for human-ai conversations. *arXiv preprint arXiv:2312.06674*, 2023. 13

Swapnaja Achintalwar, Adriana Alvarado Garcia, Ateret Anaby-Tavor, Ioana Baldini, Sara E. Berger, Bishwaranjan Bhattacharjee, Djallel Bouneffouf, Subhajit Chaudhury, Pin-Yu Chen, Lamogha Chiazor, Elizabeth M. Daly, Rogério Abreu de Paula, Pierre Dognin, Eitan Farchi, Soumya Ghosh, Michael Hind, Raya Horesh, George Kour, Ja Young Lee, Erik Miehling, Keerthiram Murugesan, Manish Nagireddy, Inkit Padhi, David Piorkowski, Ambrish Rawat, Orna Raz, Prasanna Sattigeri, Hendrik Strobelt, Sarathkrishna Swaminathan, Christoph Tillmann, Aashka Trivedi, Kush R. Varshney, Dennis Wei, Shalisha Witherspooon, and Marcel Zalmanovici. Detectors for safe and reliable llms: Implementations, uses, and limitations, 2024. 13

Adam Tauman Kalai and Santosh S Vempala. Calibrated language models must hallucinate. *arXiv preprint arXiv:2311.14648*, 2023. 14

Andrew Lee, Xiaoyan Bai, Itamar Pres, Martin Wattenberg, Jonathan K Kummerfeld, and Rada Mihalcea. A mechanistic understanding of alignment algorithms: A case study on dpo and toxicity. *arXiv preprint arXiv:2401.01967*, 2024. 14

Rafael Rafailov, Archit Sharma, Eric Mitchell, Stefano Ermon, Christopher D Manning, and Chelsea Finn. Direct preference optimization: Your language model is secretly a reward model. *arXiv preprint arXiv:2305.18290*, 2023. 14

Boyi Wei, Kaixuan Huang, Yangsibo Huang, Tinghao Xie, Xiangyu Qi, Mengzhou Xia, Prateek Mittal, Mengdi Wang, and Peter Henderson. Assessing the brittleness of safety alignment via pruning and low-rank modifications. *arXiv preprint arXiv:2402.05162*, 2024. 14

Behnam Neyshabur, Srinadh Bhojanapalli, David McAllester, and Nati Srebro. Exploring generalization in deep learning. *Advances in neural information processing systems*, 30, 2017. 15

Wesley J Maddox, Pavel Izmailov, Timur Garipov, Dmitry P Vetrov, and Andrew Gordon Wilson. A simple baseline for bayesian uncertainty in deep learning. *Advances in neural information processing systems*, 32, 2019. 15

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*, 2023. 15

Robert D Gordon. Values of mills' ratio of area to bounding ordinate and of the normal probability integral for large values of the argument. *The Annals of Mathematical Statistics*, 12(3):364–366, 1941. 16

Appendix

A. Illustration of our framework

An illustration of our framework is depicted in Figure 1.

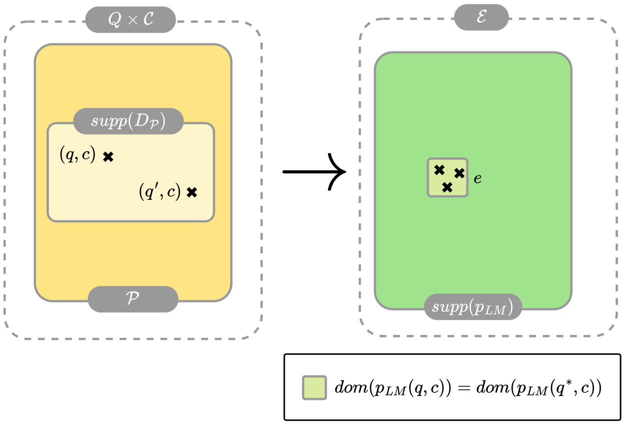

Figure 1: **Our framework in a nutshell:** We define a language model, p_{LM}: 🟨 → 🟩, as a map from prompts to a distribution over a subset of all possible explanations $\mathcal{E}$. To later be able to bound the strength of the adversarial attacker, we split the text inputs into concepts and queries (q, c). We assume that (i) the text corpus only covers a part of the domain of the LM: $\text{supp}(D_{\mathcal{P}}) \subsetneq \text{dom}(p_{LM})$, (ii) the size of the domain of the output distribution, denoted $|\text{dom}(p_{LM}(q, c))|$, is small compared to the size of $\mathcal{E}$, and (iii) only concepts determine the output (see 🟩).

A depiction of jailbreaking is in Figure 2.

B. Related work

In this section, we provide a review of the current literature on LLM jailbreaking.

B.1. Jailbreak methods

In this section, we summarize existing jailbreaking methods.

Baseline and pioneers Autoprompt (Shin et al., 2020), a baseline method for optimizing in the token space w.r.t. a certain objective, approximates coordinate ascent by first ranking all tokens using an approximate objective, and then compute the exact value on them. The approximation is carried out by a single step of gradient computation. Jones et al. (2023) propose Autoregressive Randomized Coordinate Ascent (ARCA) to generate (input, output) pairs that include certain harmful info or satisfy a fixed format requirement. Token level optimization is carried out with linear approximation on GPT-2. GBDA (Guo et al., 2021) study adversarial attack on text classification problems, by optimizing the continuous relaxation of the autoregressive

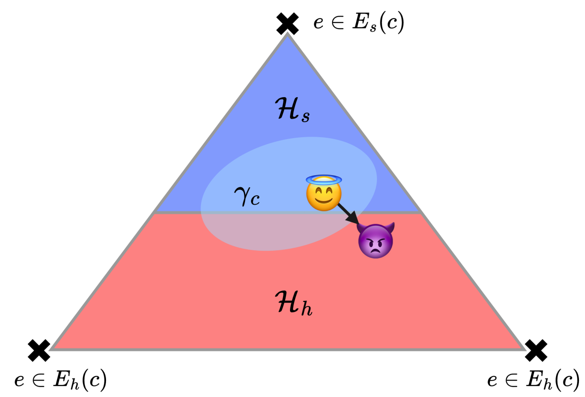

Figure 2: Conceptual illustration of our framework for jailbreaking introduced in Section 4, with a fixed harmful concept c. The triangle represents the probability simplex. This figure showcases a typical successful jailbreaking attempt by the adversary: although safety alignment makes the sampled LM safe under the direct prompt input, the adversary is able to move the output to the harmful zone $\mathcal{H}_h$ by manipulating the query q.

sampling probability matrix. In late 2022, among social media, users misalign GPT-3 via prompt injection. Perez and Ribeiro (2022) study how this be done by adversaries. They successfully manipulate the model to output a given harmful string and leak the system prompt. In early 2023, an empirical study was carried out by Liu et al. (2023a) to measure the result of prompt engineering for breaking ChatGPT safeguards. Shen et al. (2023) collect jailbreaking prompts from multiple platforms on the internet, analyze these data, create a harmful question set, and identify some typical harmful prompts that are effective at that moment. Later, the Greedy Coordinate Gradient (GCG) method (Zou et al., 2023), a strong *white-box* attack variant of AutoPrompt (Shin et al., 2020) was proposed. Wei et al. (2023a) categorizes two general modes of jailbreaking: *competing objective* and *mismatched generalization*.

LLM automation and suffix-based attacks Liu et al. (2023b) propose AutoDAN, that relies on genetic algorithms, with the requirement of manual prompts for conducting mutation and crossover on the *paragraph and sentence level*. The jailbreaking prompts generated by AutoDAN are semantically plausible, unlike the suffix generated by GCG. As a comparison, Lapid et al. (2023) use genetic algorithm for *black-box* universal adversarial suffix generation. Chao et al. (2023) propose another LLM-based jailbreak automation algorithm, where an LLM judge is built to assign a safety score to a given output, while the attacker is enforced (via a page-long prompt) to improve the quality of jailbreaking prompts from multiple perspectives. Zhu et al. (2023) propose another AutoDAN method that explores the balanced loss between jailbreak loss (log probability on the harmful string, as used in Zou et al. (2023)) and the plausibility loss (log probability over the adversarial suffix), aiming at improving interpretability. Li et al. (2024a) uses genetic algorithm to search with similarty measure and initialize with paraphrasing. Its performance is claimed to be better than AutoDAN-GA. Deng et al. (2023a) investigate the possible defensive tricks in proprietary LLMs, and propose a pipeline[4] for automated jailbreaking using a fine-tuned LLM. Yu et al. (2023) propose GPTFuzzer, essentially a genetic algorithmic framework for jailbreaking. Their work's difference between AutoDAN is that it has a pool of "seeds", a.k.a. templates for transforming the harmful prompt, and the mutation is done on the template level. Ding et al. (2023) propose automating attack via LLM *prompt rewriting* and *scenario nesting*. The latter consists of code completion, table filling and text continuation, since the authors regard these as align with training objectives well, and are suitable for LLMs to complete the task. Mehrotra et al. (2023) combine Automation used in Chao et al. (2023) and tree-of-thought (Yao et al., 2024), create interpretable prompts in a *black-box* manner. Li et al. (2023a) propose DeepInception, and use *nested, imaginary* scenario to induce harmful content. Li et al. (2024b) propose DrAttack, which camouflages a query's malicious intent through semantic decomposition, by constructing a parsing tree and split the original prompt into different segmentations. Wang et al. (2024a) draw inspiration from the self-perception theory from psychology to design a prompt modification pipeline on gradually persuading the LM to be jailbroken. Paulus et al. (2024) propose AdvPrompter, where the authors train a language model as a *suffix generator* to speed up LLM attack.

Manipulating the decoding process Huang et al. (2023) find the method of changing the generating hyperparameters (*i.e.,* p of top-p sampling, the temperature T, and k of top-k sampling) of safety-aligned LLMs suffices for obtaining harmful outputs when the user is able to manipulate the system prompt and input configurations. Zhang et al. (2023a) directly manipulate the output generation probability by enforcing an affirmative prefix, and reversing the negation words if they appear in a pre-defined vocabulary (*e.g.,* sorry → glad). Zhao et al. (2024) assume access to the decoding distribution of a LM. They use two small LMs, a safe one and a harmful one, to manipulate the decoding ratio of the large safe LM for jailbreaking. The key insight is the decoding distribution between the safe model and the harmful model only differs significantly for the first tens of tokens.

[4] Including query rewriting, training and fine-tuning

Fine-Tuning alone suffices Yang et al. (2023) show that fine-tuning on as few as 100 harmful example pairs suffices for turning the LLaMa-chat models (and some other <70B LMs) into malicious counterparts. Zhan et al. (2023) fine-tune GPT-4 on harmful data, and find the fine-tuned models escape previous safety constraints while maintaining usefulness. Qi et al. (2023a) find fine-tuning alone, even on benign data, leads to safety degradation using LLaMa and GPT-3.5-Turbo. Fine-tuning on harmful data (with less than 5 gradient steps) will cause the model to be completely harmful, while tuning on identity-shifting data could make the LM fully obedient.

Low-resource language and cipher Yong et al. (2023); Deng et al. (2023b) explore the difference in languages when encountering the same harmful query, and find a direct translation to low resource languages will lead to higher risk, and Deng et al. (2023b) additionally find when combined with sophisticated methods, the drawback of low-resource languages disappear. Yuan et al. (2023) use cipher encoding and decoding to break LLMs. Smaller scale models are immune from such attacks, while the smartest GPT-4 encountered the highest risk.

Vision-language model attacks Besides pure LLM, some research works move a step forward, utilizing images for breaking vision-language models (VLMs). Shayegani et al. (2023) explore multimodal attack on VLM via embedding space feature matching. Qi et al. (2023b) generate adversarial examples via maximizing the conditional probability of a harmful *corpus, i.e.,* the sum of log probabilities over all outputs, and use the final image with harmful query for jailbreaking. Carlini et al. (2023) generate adversarial example for a *fixed harmful content*, and find no matter what input prompt is given to the VLM, it will respond with the target harmful string. Maus et al. (2023) propose a black-box attack on manipulating the generated image with modified adversarial prompt.

Misc Wei et al. (2023b); Wang et al. (2023a) explore in-context learning for attack and defense. The attack is weak since it could only break Vicuna (Chiang et al., 2023) and can be defended by in-context safe examples. Later, this method is scaled-up to significantly improve strength for breaking guardrails of large, state-of-the-art models (Anil et al., 2024). An early work in February 2023 (Kang et al., 2023) adopts obfuscation (including synonym and typos), code injection and virtualization to successfully jailbreak ChatGPT. Shah et al. (2023) illustrate in-context automated persona modulation attack for large-scale LLMs and Vicuna. Zeng et al. (2024a) consider the more broadly perspective of *persuasion*: they train a persuasive paraphraser based on a fine-grained taxonomy of persuasion techniques. Detailed ablation on attack effectiveness is studied. Guo et al. (2024) focus on stealthiness and controllability. They notice the constraints applied to the jailbreaking prompts (*e.g.,* fluency) are exactly the targets of the controllable text generation problem. Thus, they adopt the Energy-based Constrained Decoding with Langevin Dynamic (COLD) (Qin et al., 2022) on output logits. Forming each constraint as well as the task of jailbreaking as an energy function over logits, the Langevin Dynamic is used for finding a good logit distribution, and the decoding technique in Qin et al. (2022) is used for output generation. Banerjee et al. (2024) introduce a dataset TECHHAZARDQA, compare direct query v.s. pseudo-code format, and find the latter induces higher risk. Mangaokar et al. (2024) considers a type of adaptive attack against *checking-based defense*, that appends a universal adversarial prefix into the jailbreaking prompt to make the guard model always output "safe", and thus making the detector fails to detect harmful information. Lv et al. (2024) propose Code Chameleon, which contains multiple encryption and decryption methods defined by python functions, that transforms the harmful prompt into specific predefined form to jailbreak LLMs. Sadasivan et al. (2024) speed up the computation of GCG (Zou et al., 2023) to make it possible to launch on a single GPU. Geiping et al. (2024) build a taxonomy on risks beyond jailbreaking, and coerce the LLM to provide certain outputs by optimizing a set of tokens via GCG. Ren et al. (2024) propose CodeAttack that use code templates to query the output out instead of using natural language directly, and obtain descent results.

B.2. Defense methods

Up to now, no universal defensive strategy as adversarial training (Madry et al., 2018) for adversarial attacks / differential privacy (Abadi et al., 2016) for membership attacks exists as a gold standard. In general, we can classify the methods into three typical types: alignment, read-teaming, and algorithmic proposals.

Alignment The target of alignment is to push the output of language models be aligned to human values. Regarding safety, the goal is to avoid outputting harmful information. RLHF is widely adopted in these methods (Ziegler et al., 2019; Ouyang et al., 2022; Bai et al., 2022a; Korbak et al., 2023). Variants like RLAIF also have been proposed recently (Bai et al., 2022b; Lee et al., 2023).

Red teaming This term is populated as specifically dealing with harmful info on dataset curation, used together with RLHF (Ganguli et al., 2022; Perez et al., 2022; Casper et al., 2023; Hong et al., 2023; Samvelyan et al., 2024).

Next, we proceed to defensive algorithm proposals. We classify existing defensive strategies in the following categories.

Defense against suffix-based attacks. Alon and Kamfonas (2023) notice the messy nature of the suffix generated by GCG, and propose to use a perplexity (PPL) filter on input prompts. They also explore using a LightGBM (Ke et al., 2017) with 2 features (PPL, prompt length) to filter harmful prompt, and show it does better than naive PPL thresholding. The PPL-based filter does not succeed with human-crafted jailbreaks. Jain et al. (2023) explore many concerning viewpoints, including self-PPL filtering, paraphrasing the input prompt, and re-tokenization since many LLMs' tokenizers are based on Byte-Pair Encoding (BPE). All methods are successful in regards of defending against suffix-based attacks. They also explore the simplest form of adversarial training. Robey et al. (2023) propose to perturb the input token string by random replacement/erasement/insertion, and finally perform a majority vote in the end. Cao et al. (2023) judges whether an input prompt is safe or not by estimation with Monte Carlo, when randomly dropping a fraction of tokens, using the LLM itself. Kumar et al. (2023) try to perform "certified" safety against harmful prompts, by erasing tokens and set the original prompt as harmful if at least one of these erased prompts lead to a harmful response, or be classified as harmful by a DistillBERT-based classifier.

System prompt defense. We could modify the input prompt for jailbreaking; and several works explore if we can apply similar methods to system prompts to defend against such attacks. Xie et al. (2023) propose "self-reminder", *i.e.,* appending a reminding prompt within the system prompt for defense. The attacks are collected from the JailbreakChat collection, and this strategy is effective for defending against them. Zheng et al. (2024) advocate for finding a good system prompt automatically, by investigating the representational difference between safe and harmful queries, and optimizing the safety prompts along the harmful refusal direction in the representation space. One intriguing takeaway is harmful / harmless queries can be distinguished in the representation space, different from the adversarial examples in vision. Zhou et al. (2024a) also optimize the safe system prompt, but in a more "adversarial training" fashion, that apply jailbreak algorithms with current safe prompts first and then find good replacement candidates in the same way as done by Zou et al. (2023). Concurrently, Mo et al. (2024) propose prompt adversarial tuning, where an adversarial suffix is assumed, while a safe system prompt is jointly optimized with this suffix, *with an additionally constructed benign loss* to improve helpfulness under normal queries. Zhang et al. (2023b) propose the idea of "goal prioritization", either without training (append prioritize safety than helpfulness and in-context examples to the system prompt) or with training (generate data pairs of prioritizing safety or helpfulness, finetune, and append prioritize safety prompt into system prompt). The former is effective for large-scale LLMs, while the latter improves safety of LLaMa-chat models. Zhou et al. (2024b) propose in-context adversarial game, where an attack LLM and a defense LLM interact on exchanging insights on successful jailbreaks, and defend by improving the system prompt. Zou et al. (2024) give the result that system prompt matters for jailbreaking, and shows conducting GA-based search over it could improve safety.

Checking-based, decoding-based, and Misc. Helbling et al. (2023) generate responses first, and then use the LLM itself to examine whether the output is harmful or not. They find such simple self-examination is powerful since the TPR reported can be up to ~ 1.00. Wang et al. (2023b) propose to (1) tune the LM to enhance its capability on discriminating harmful / harmless content; (2) tune the LM to make it able to tag its own response; and (3) rewrite response if output is harmful. Li et al. (2023b) propose to suppress the attack performance by iteratively rewinding and re-examining the generated output. The method does not work well with small models, but works pretty fine with large (open-source) models. They find the strategy can improve generalization as well. Xu et al. (2024) train a safer model first, and use normalized $p_{\text{attacked}} + \alpha(p_{\text{safer}} - p_{\text{attacked}})$ over top-k shared tokens for decoding to enhance safety. Hasan et al. (2024) show with original Wanda pruning (Sun et al., 2023), the LLM can help resist direct jailbreaking prompts, *e.g.,* with role-playing attacks. Pi et al. (2024) propose MLLM-Protector on safeguarding Visual LLMs by checking the output and then detoxifying the content. Zhang et al. (2024) perform intention analysis on the input, and enforce the model generate policy-aligned outputs both by prompting. Wang et al. (2024b) propose backtranslation that guesses the input prompt directly, and reject if it is harmful. Kim et al. (2024) propose self-refinement which consists of generating a feedback and then refine the response to avoid harmful info output. They find using additional JSON and code formatting would improve safety. Zeng et al. (2024b) propose AutoDefense, which utilizes multiple agents on analyzing prompt, response and intention, to defend against attacks. Hu et al. (2024) propose Gradient Cuff, a sampling-based gradient-norm defense method, by rejecting those input prompts with large gradient norm on top of a majority-vote based filtering. Ji et al. (2024) propose a method similar to Robey et al. (2023), but for semantically-meaningful attacks, that paraphrases the input according to several criteria and conduct a majority vote for judging.

Several company-centered products also fall into this regime. For example, LLaMa-Guard (Inan et al., 2023) is trained on toxic data such that it is able to discriminate unsafe user prompts and outputs, respectively. IBM also propose a framework on constructing and deploying safeguard detection modules, and releases the details in a technical report (Achintalwar et al., 2024).

B.3. Theory and experimental understanding

Wolf et al. (2023) assumes the decomposability of LLM output into a good and bad component, and show possible jailbreaking in theory by prompting the model with a sufficiently long input. Kalai and Vempala (2023) use statistical tools to prove hallucination for calibrated LMs. Lee et al. (2024) study the representation in GPT-2. They train a base classifier for toxicity, and use the linear weight as a proxy of toxic vector. They find there are value vectors close to the toxic vector itself, that are not suppressed by DPO tuning (Rafailov et al., 2023). Wei et al. (2024) use pruning and low-rank analysis on safe LM, and find (1) safe neurons and useful neurons are sparse; pruning the safe neurons or removing the safe ranks away degrades safety a lot, and (2) fixing the safe neurons in fine-tuning does not maintain safety.

C. Proof of Theorems

C.1. Proof of PAC-Bayesian bounds

Definition C.1. *(Bounded Difference) A function $f : \mathcal{X}^n \to \mathbb{R}$ is said to have bounded difference property w.r.t. a collection of constants $c_1, \cdots, c_n$, iff*

$$\sup_{x_1, x_2, \dots, x_n, x_i^{'}} |f(x_1, x_2, \cdots, x_n) - f(x_1, x_2, \cdots, x_{i-1}, x_i^{'}, \cdots, x_n)| \\ \leq c_i, \forall i \in [n].$$

Lemma C.1. *(Hoeffding's Lemma) for random variable $X \in [a, b]$ with probability 1, the following holds:*

$$\mathbb{E}[\exp(\lambda X)] \leq \exp(\lambda \mathbb{E} X + \frac{\lambda^2 (b-a)^2}{8}).$$

Lemma C.2. *(Hoeffding's Lemma, Multivariate) for random variables $Z = f(x_1, \cdots, x_n)$ where f has the bounded difference property, the following holds:*

$$\mathbb{E}[\exp(\lambda(\mathbb{E} Z - Z))] \leq \exp(\frac{\lambda^2 \sum_{i=1}^n c_i^2}{8}).$$

Note that substituting Z with $\hat{R}_S(LM)$ is valid.

Lemma C.3. *Empirical Loss defined in Definition 3.1 satisfies the bounded difference condition with constant $c = 1, \forall i$.*

We are ready to present the proof of Theorem 1.

Proof. Starting with the above lemma, we know

$$\mathbb{E}_S[\exp(\lambda(R(LM) - \hat{R}_S(LM)))] \leq \exp(\frac{\lambda^2 c^2}{8n}).$$

The above result holds for a manually picked LM. With an overall average over the prior π we have

$$\mathbb{E}_{LM \sim \pi} \mathbb{E}_S[\exp(\lambda(R(LM) - \hat{R}_S(LM)))] \leq \exp(\frac{\lambda^2 c^2}{8n}).$$

Apply Fubini's theorem (note that π is independent of S):

$$\mathbb{E}_S \mathbb{E}_{LM \sim \pi}[\exp(\lambda(R(LM) - \hat{R}_S(LM)))] \leq \exp(\frac{\lambda^2 c^2}{8n}).$$

Define $Y = \mathbb{E}_{LM \sim \pi}[\exp(\lambda(R(LM) - \hat{R}_S(LM)))]$, a random variable depends on S. Obviously $Y \geq 0$. Thus, with Markov's inequality:

$$\mathbb{P}[Y \geq \frac{1}{\delta} \mathbb{E}_S Y] \leq \delta.$$

Equivalently, with probability at least $1 - \delta$, we have

$$Y \leq \frac{1}{\delta} \exp[\frac{\lambda^2 c^2}{8n}].$$

Since we have assumed π, ρ share the same support, using Radon-Nykodim derivative to change the expectation with respect to π to with respect to ρ, we have

$$\mathbb{E}_{LM \sim \rho}\left[\frac{d\pi}{d\rho} \exp(\lambda(R(LM) - \hat{R}_S(LM)))\right] \leq \frac{1}{\delta} \exp[\frac{\lambda^2 c^2}{8n}].$$

Taking logarithm and applying Jensen's Inequality we know

$$\mathbb{E}_{LM \sim \rho}\left[\frac{d\pi}{d\rho} + \lambda(R(LM) - \hat{R}_S(LM))\right] \leq \log \frac{1}{\delta} + \frac{\lambda^2 c^2}{8n}.$$

Incorporating $c = 1$, noticing $\frac{d\rho}{d\pi} = (\frac{d\pi}{d\rho})^{-1}$ we could rewrite the inequality as

$$\mathbb{E}_{LM \sim \rho}\left[(R(LM) - \hat{R}_S(LM))\right] \leq \frac{1}{\lambda}\left(\mathrm{KL}[\rho || \pi] + \log \frac{1}{\delta}\right) + \frac{\lambda}{8n}.$$

Finding λ that minimizes the term on right hand side gives us the ϱ term.

When $D_{\mathcal{P}}$ allows for a decomposition into mixture components, noticing the linearty of expectation, the bound can be re-written as

$$\begin{aligned} &\alpha \mathbb{E}_{LM \sim \rho}[\mathbb{E}_{(q,c) \sim D_{\mathcal{P}_h}} \ell_{\mathrm{TV}}(p_{LM}, (q, c))] \\ &\quad + (1-\alpha) \mathbb{E}_{LM \sim \rho}[\mathbb{E}_{(q,c) \sim D_{\mathcal{P}_s}} \ell_{\mathrm{TV}}(p_{LM}, (q, c))] \\ &\quad \leq \varrho + \mathbb{E}_{LM \sim \rho}[\hat{R}_S(p_{LM})]. \end{aligned}$$

which leads to

$$\begin{aligned} &\mathbb{E}_{LM \sim \rho}[\mathbb{E}_{(q,c) \sim D_{\mathcal{P}_h}} \ell_{\mathrm{TV}}(p_{LM}, (q, c))] \leq \\ &\qquad \frac{1}{\alpha}[\varrho + \mathbb{E}_{LM \sim \rho}[\hat{R}_S(p_{LM})]]. \end{aligned} \tag{1}$$

□

C.2. An estimation on the non-vacuousness of the PAC bound

We give an estimation of the term appears in our PAC bound, ϱ, and state that it is non-vacuous.

The numerator. We follow Neyshabur et al. (2017) to instantiate the term in the simplest setup. Assume π, ρ are defined over the parameter space of a given LM, with K parameters. Assume w is a set of weights learned from the pretraining corpus. Let the prior π be the zero-mean multivariate Gaussian, whose entry-wise variance is related to the magnitude of the weight: $\sigma_i = \beta |w_i|$, and ρ be a Gaussian with the same anisotropic variance centered around w. We argue though simple, both settings are practical, since Gaussian initialization is common for model training, and the SWA-Gaussian algorithm (Maddox et al., 2019) utilizes such Gaussian posterior. Under this setup, the KL goes as $\sum_i \frac{w_i^2}{2\sigma_i^2} = O(K)$. Specifically, taking $\beta = \frac{\sqrt{2}}{2}$ makes the term exactly K. Current language models often possess millions, or billions, of parameters, namely, $K \sim [10^6, 10^9]$.

The denominator. To estimate the number of unique direct prompts in the training corpus, it is important to notice that the dataset does not only consist of (q, c) prompts but also e explanations. Thus, we need to estimate the *average token length (ATL)* associated with each unique prompt $x = (q, c)$. For each unique prompt x, aside from its own token length $l(x)$, there will be a collection of explanations $\{e_i\}_{i=1}^{N(x)}$, with expected token length of each associated explanation $l(e)$. We have

$$\mathbb{E}ATL = \mathbb{E}_{x \sim D_{\mathcal{P}}} N(x) \times [l(x) + l(e)].$$

Fact. Given a prompt x, the larger the expected length of the prompt itself and explanation ($l(x) + l(e) \uparrow$), the larger the expected *number of explanation elements* ($N(x) \uparrow$), and the smaller the number of such prompts ($D_{\mathcal{P}}(x) \downarrow$), appearing in the training corpus. The former comes naturally due to the composability of natural language: the longer the text fragment, the more equivalent text fragments in expectation, while the latter is reflected by the spirit of the widely accepted Zipf's law.

Inspired by the fact, we assume prompts are categorized by the quantity of $l(x) + l(e)$, namely, for all prompt x, $N(x)$ is a function of $l(x) + l(e)$. Moreover, the complete data generation process is decomposed into i) sample a value of $l(x) + l(e)$ out, and then ii) sample a unique prompt from the set decided by this specific $l(x) + l(e)$ value, and iii) generate $N(x)$ explanations.

Step i). Use the fact: the larger the expected length of the output explanation, the smaller the probability that such a prompt appears in the training corpus. We assume step i) follows a (cut-off) zeta distribution. Specifically, for a random prompt x,

$$p(l(x) + l(e) = k) \propto k^{-s}, \forall k \geq k_0.$$

When $k_0 = 1$, we resume the zeta distribution with coefficient s.

Step ii). We assume each prompt following this step is unique.

Step iii). Use the fact: the larger the expected length of the output explanation, the larger the expected *number of explanation elements* in the training corpus. We assume a power law scaling on N, with a constant $t > 1$, such that

$$N(l(x) + l(e) = k) = k^{t-1}.$$

Thus, the average token length writes

$$\begin{aligned}\mathbb{E}ATL &= \sum_k p(l(x) + l(e) = k) \times k \times N(l(x) + l(e) = k) \\ &= \frac{\zeta(s-t) - \sum_{i=1}^{k_0-1} i^{-(s-t)}}{\zeta(s) - \sum_{i=1}^{k_0-1} i^{-s}}.\end{aligned}$$

where $\zeta(s) = \sum_{i \in \mathbb{Z}^+} i^{-s}$ is the Riemann zeta function.

For example, take $s = 4, t = 2$. With $k_0 = 1$, the ATL would be 1.52, while with $k_0 = 10$, the ATL becomes 272. These results translate into an estimation of unique prompts as n_{tokens}/ATL. With current SOTA LM, the pretraining corpus often includes (tens of) trillions of tokens ($> 10^{12}$), thus $n > 10^{10} > K$ can be safely assumed $\Rightarrow \varrho < 1$.

α **constant.** According to LLaMa-2 report (section 4.1, Figure 13) (Touvron et al., 2023), approximately 0.2% of the documents in their training corpus is labeled as harmful. However, we argue this is indeed an extremely **loose** lower bound for α, due to the estimation strategy used in their paper. Given a document, they use a binary classifier on harmfulness over *each single line* (1 means harmful and 0 otherwise), and assign the *average* score to the document. 0.2% is the ratio of documents *with score* ≥ 0.5. Take the example of "`How to build a bomb`". The chemical reaction parts will not be counted as harmful, and thus this estimation strategy could judge a completely harmful explanation as harmless. Thus, it is reasonable to assert α is not too small, though with current literature we are not capable of raising an accurate estimation on it.

C.3. Validity of Assumption 4.1

The $O(1)$ statement is reasonable, because harmful explanations are usually long text fragments that allow for many alternative formulations. The assumption can be broken down into two components: (1) within the support of the output distribution, only occasional instances of unrelated explanations exist; (2) the process of aligning the model towards safety **does not eliminate the harmful explanations** acquired during the pretraining phase. For part (1), similar to the example we gave above, under normal circumstances, we do not expect the explanation "`Paris`" to appear in $\text{dom}(p_{LM}(q, c))$ given (q, c) as "`How to build` a bomb". As for part (2), though seemingly surprising, evidence with a series of current state-of-the-art LMs can be experimentally validated (Huang et al., 2023), where diverse, harmful explanations are extracted by simply manipulating the decoding process using direct prompts.

C.4. Proof of jailbreaking

Before proceeding to the proof, we list necessary definitions and lemmas as follows.

Lemma C.4. *(Volume of n-simplex)*[5] *For any dimension n, the volume of the n-element probability simplex: Δ^{n-1}, in the $n-1$-dimensional space is*

$$\frac{\sqrt{n}}{(n-1)!}.$$

We define the projected probability simplex as follows.

Definition C.2. *(Projected probability simplex) Given Δ^{n-1}, the corresponding projected probability simplex, Δ_p^{n-1}, is defined as a subset of $\mathbb{R}^{n-1}$: $\{x \in \mathbb{R}^{n-1} | \sum_{i=1}^{n-1} x_i \leq 1, \forall i \in [n-1]\}$.*

An illustration of Δ^{n-1} and Δ_p^{n-1}. For example, take $n = 3$. The probability simplex with $n = 3$ elements is a triangle whose (euclidean) side length is $\sqrt{2}$ with vertices $(1,0,0), (0,1,0), (0,0,1)$. Then its volume in the 2-dimensional space, *i.e.,* its area, is $\frac{\sqrt{3}}{2}$. The corresponding projected probability simplex is the triangle between the $X-Y$ axis, with vertices $(1,0), (0,1), (0,0)$.

A direct lemma that connects the probability simplex and the projected probability simplex is given below.

Lemma C.5. *(Transformation of probability simplex) Given a proper probability density function $\nu(x)$ defined on Δ_p^{n-1}, it is equivalent to the distribution defined on Δ^{n-1} with density $\frac{\nu(x)}{\sqrt{n}}$: $\forall A \in Borel(\Delta_p^{n-1})$, let $B = \{x \in \Delta^{n-1} : x_{1:n-1} \in A\}$. Then $\int_A \nu(x)dx = \int_B \frac{\nu(x)}{\sqrt{n}}dx$. Specifically, this implies $\frac{volume(A)}{volume(\Delta_p^{n-1})} = \frac{volume(B)}{volume(\Delta^{n-1})}$.*

Proof. Consider a translation on Δ^{n-1} with $x_n = -\sum_{i=1}^{n-1} x_i$ which does not affect its the volume and shape. The mapping: $\Delta_p^{n-1} \rightarrow \text{translated}\Delta^{n-1}$ is an affine transformation with matrix

$$T = \begin{pmatrix} 1 & 0 & \cdots & 0 \\ 0 & 1 & \cdots & 0 \\ \cdots & \cdots & \cdots & \cdots \\ -1 & -1 & \cdots & -1 \end{pmatrix}_{n \times (n-1)}$$

Thus, any area under this transformation is scaled by $\sqrt{\det T^\top T} = \sqrt{n}$: a constant. The lemma follows directly after this conclusion. □

We use $U(\cdot)$ to denote the uniform distribution over Δ^{n-1}: $U(x) = \frac{(n-1)!}{\sqrt{n}}, \forall x \in \Delta^{n-1}$. We use the notation $\text{vol}[S] = \int_S 1 ds$ to represent the volume of a given subset $S \subset \Delta^{n-1}$, and use $\text{rvol}[S]$ for the relative volume (w.r.t. the underlying n-simplex) of S, *i.e.,* $\text{rvol}[S] := \frac{\text{vol}[S]}{\text{vol}[\Delta^{n-1}]} = \int_S U(x)dx$. We also use $n = |E(c)|$ from now on. We use the vector x to denote (with the slight abuse of notation we have mentioned) $p_{LM}(q, c)$ on the output simplex.

Lemma C.6. *(Gaussian cdf Tail Bound, Gordon (1941)) Denote $\phi(\cdot)$ as the standard Gaussian pdf. When $x > 0$,*

$$\frac{x}{x^2+1}\phi(x) = \frac{x}{x^2+1}\frac{e^{-x^2/2}}{\sqrt{2\pi}} \leq 1 - \Phi(x) \leq \frac{e^{-x^2/2}}{\sqrt{2\pi}x} = \frac{1}{x}\phi(x).$$

Now we are ready to give the proof of Theorem 2.

Proof. Let $|E_h(c)| = n_0$ and denote $|E_h(c)| + |E_s(c)| + |E_n(c)| = n$. Without loss of generality, we define the first $n_0 = |E_h(c)|$ elements as the harmful explanations. Let the thresholding constant be p. That is, we define the harmful zone $\mathcal{H}_h$ as $\{x \in \Delta^{n-1} | \sum_{i=1}^{n_0} x_i \geq p\}$. To compute the relative volume of $\mathcal{H}_h$ in Δ^{n-1}, we could instead operate on the projected probability simplex Δ_p^{n-1} introduced in Definition C.2, and compute the relative volume of the projected $\mathcal{H}_h$: $\mathcal{H}_{h,p} := \{x \in \Delta_p^{n-1} | \sum_{i=1}^{n_0} x_i \geq p\}$. Note that $\Delta_p^{n-1} \subset \mathbb{R}^{n-1}$. We derive its expression as follows.

$$\begin{aligned}
&\text{volume}[\mathcal{H}_{h,p}^C] = \text{volume}[\{x \in \Delta_p^{n-1} | \sum_{i=1}^{n_0} x_i \leq p]\} \\
&= \int_0^p dx_1 \int_0^{p-x_1} dx_2 \cdots \int_0^{p-\sum_{i=1}^{n_0-1} x_i} dx_{n_0} \\
&\times \int_0^{1-\sum_{i=1}^{n_0}} dx_{n_0+1} \cdots \int_0^{1-\sum_{i=1}^{n-2} x_i} dx_{n-1} \\
&= \int_0^p dx_1 \int_0^{p-x_1} dx_2 \cdots \int_0^{p-\sum_{i=1}^{n_0-1} x_i} dx_{n_0} \\
&\times \left[\frac{1}{(n-n_0-1)!} (1 - \sum_{i=1}^{n_0} x_i)^{n-n_0-1} \right] \\
&= \int_0^p dx_1 \int_0^{p-x_1} dx_2 \cdots \int_0^{p-\sum_{i=1}^{n_0-2} x_i} dx_{n_0-1} \\
&\times \frac{1}{(n-n_0)!} \left[(1 - \sum_{i=1}^{n_0-1} x_i)^{n-n_0} - (1-p)^{n-n_0} \right] \\
&= \cdots \\
&= \frac{1}{(n-1)!}[1-(1-p)^{n-1}] - \sum_{j=1}^{n_0-1} \frac{(1-p)^{n-1-j}}{j!(n-1-j)!} p^j
\end{aligned} \tag{2}$$

[5] See https://en.wikipedia.org/wiki/Simplex#Volume.

Thus, the relative volume of $\mathcal{H}_h$ can be written as

$$\begin{aligned} \text{rvol}[\mathcal{H}_h] &= 1 - \frac{\text{volume}[\mathcal{H}_{h,p}^C]}{\text{volume[projected probability simplex]}} \\ &= (1-p)^{n-1} + \sum_{j=1}^{n_0-1} \frac{(n-1)!(1-p)^{n-1-j}}{j!(n-1-j)!} p^j \\ &= \sum_{j=0}^{n_0-1} p^j (1-p)^{n-1-j} \binom{n-1}{j}. \end{aligned} \tag{3}$$

Which is precisely the binomial distribution formula. With the Central Limit Theorem, when $n \gg O(1)$, we know the binomial distribution can be well approximated via the normal distribution as follows:

$$f(x) = \binom{n}{x} p^x (1-p)^{n-x} \xrightarrow{d} \mathcal{N}(np, np(1-p)). \tag{4}$$

Thus, denote $\phi_{(n-1),p}(x)$ as the pdf of Gaussian variable with mean $(n-1)p$, variance $(n-1)p(1-p)$, the rvol term above can be estimated as follows:

$$\begin{aligned} \sum_{j=0}^{n_0-1} p^j (1-p)^{n-1-j} \binom{n-1}{j} &\asymp \int_{-\infty}^{n_0-1} \phi_{(n-1),p}(x) dx \\ &= \Phi\left[\frac{n_0 - 1 - (n-1)p}{\sqrt{(n-1)p(1-p)}}\right] \\ &= \Phi\left[\frac{|E_h(c)| - 1 - (n-1)p}{\sqrt{(n-1)p(1-p)}}\right]. \end{aligned} \tag{5}$$

We use $a = \frac{|E_h(c)|-1-(n-1)p}{\sqrt{(n-1)p(1-p)}}$. Consider an adversary with budget ϵ under ℓ_p or Jensen-Shannon Divergence (JSD) / Total Variation (TV) capability. Since $||x||_1 \geq ||x||_p, \forall p \geq 1$ as well as $||x||_1 \geq 2\text{JSD}(x), ||x||_1 \geq 2\text{TV}(x)$, we know $\mathcal{H}_h(\epsilon, \ell_1) \subset \mathcal{H}_h(\epsilon, d)$ for all d we have considered. With that ℓ_1, ϵ setup, the corresponding $\epsilon-$expansion set of $\mathcal{H}_h$ has a closed-form expression as

$$\mathcal{H}_h(\epsilon, \ell_1) = \{x \in \Delta^{n-1} | \sum_{i=1}^{n_0} x_i \geq p - \frac{\epsilon}{2}\}.$$

Similar as above, we derive the analytical solution of its relative volume associated with constant a' as:

$$\begin{aligned} a' &= \frac{|E_h(c)| - 1 - (n-1)(p - \frac{\epsilon}{2})}{\sqrt{(n-1)(p-\frac{\epsilon}{2})(1-p+\frac{\epsilon}{2})}} \\ &= a\sqrt{\frac{p(1-p)}{(p-\frac{\epsilon}{2})(1-p+\frac{\epsilon}{2})}} + \frac{\epsilon}{2}\sqrt{\frac{n-1}{(p-\frac{\epsilon}{2})(1-p+\frac{\epsilon}{2})}}. \end{aligned} \tag{6}$$

Under our framework, with $p < \frac{1}{2}$, we know $\frac{1}{4} > p(1-p) > (p - \frac{\epsilon}{2})(1 - p + \frac{\epsilon}{2}))$. Thus

$$a' > a + \sqrt{n-1}\epsilon := a_\epsilon.$$

Consider the induced distribution γ_c on the output simplex. Given an adversary with ℓ_p or JSD/TV perturbing capability, with the fixed harmful concept c, safety is guaranteed if and only if $p_{LM}(q, c)$ resides outside $\mathcal{H}_h(\epsilon, d)$. Define the area of interest, $S(d)$ as $S(d) := \Delta^{n-1} - \mathcal{H}_h(\epsilon, d)$. Thus, the probability of this event could be bounded as

$$\begin{aligned} \mathbb{P}_{x\sim\gamma_c}\mathbb{1}_{x\in S(d)} < \max_{x\in S(d)} \gamma_c(x) \int_{S(d)} 1dx &< \gamma_s \text{rvol}[S(d)] \\ < \gamma_s \text{rvol}[S(\ell_1)] &< \gamma_s(1 - \text{rvol}[\mathcal{H}_h(\epsilon, \ell_1)]) \end{aligned}$$

This gives an upper bound of

$$\gamma_s(1 - \Phi(a_\epsilon)).$$

which can be simplified when $a \geq 0$ using Lemma C.6:

$$\gamma_s\left(\frac{\phi(a_\epsilon)}{a_\epsilon}\right).$$

Thus, the probability of getting a LM instance from the preference alignment process such that it allows for successful jailbreaking on a specific harmful concept c is at least

$$1 - \gamma_s\left(1 - \Phi(a_\epsilon)\right).$$

Up to now, we have derived the result in Theorem 2. However, we can move a step further to show the decay rate on the right hand side term. It can be simplified when $a \geq 0$:

$$1 - \gamma_s\left(\frac{\phi(a_\epsilon)}{a_\epsilon}\right),$$

which finishes the proof. □