https://ieeexplore.ieee.org/abstract/document/7344789?casa_token=Y8vr7e5rYsMAAAAA:e_Oilz5WxyOFZqylzR9T_SW-w510LvvZrzZO5o0TcUdW7QJl_-NiSl_CCe1x0ttwc3ZmQ80

Inference optimization and decoding techniques against LLM attacks without sacrificing speed

1. Temperature sampling
2. Nucleus sampling
3. Speculative decoding
4. Epsilon sampling
5. Beam search
6. Ancestral sampling

1. See what works and what does not work
2. Extract a general framework that tradeoffs different methods
3. Evaluate the sensitivity of these attacks to decoding
4. Design an adaptive attack that works against sampling defenses
5. Speed up the attack optimization method - first make it faster by writing cuda kernels
6. Extend the attack to visual-language models
7. What other ways to 'align' or prevent harmful response besides RL

Use ART for generating multiple attacks - see if you combine images which succeed. Design adversarial attacks where the target is captchas or any other guardrail e.g medical image e.t.c

https://huggingface.co/datasets/project-sloth/captcha-images
https://www.kaggle.com/datasets/fournierp/captcha-version-2-images
https://www.microsoft.com/en-us/security/blog/2024/02/14/staying-ahead-of-threat-actors-in-the-age-of-ai/
https://openai.com/blog/disrupting-malicious-uses-of-ai-by-state-affiliated-threat-actors

Gotcha: transferable jailbreaks for vision-language models via captcha targets.
https://huggingface.co/datasets/hammer888/captcha-data
On The Adversarial Robustness of MultiModal Foundation Models

Captcha detector

1. Supervised - captcha vs imagenet images

Llm postprocessing defenses - Philosophy of remorse is that you have more fine grained control over the output of your LLM than over the input, and defenses of the current attack paradigms should focus more on post-processing.

Focusing on pre-processing will mean coming up with different ways to defend which is futile and can introduce errors (false positives).

1. Perplexity filtering
2. Linear probing (low-rank adapting)
3. Retrofit the RLHF guard

Your LLM sampling is already slow - adding a simple linear classifier that predicts 0, 1 based on whether the output is harmful or not is relatively cheap. You can re-use your guardrail dataset to do that, and you can cache the response so that subsequent responses are quicker.

Building linear classifiers is not Novel. We're not classifying - we're aligning

Aggregate all the harmful guardrail datasets and train one-class SVM. What are the online guarantees?
ConfAIde, AdvBench, AART, SimpleSafetyTests, BeaverTails, XSTest, DoNotAnswer, MaliciousInstructions, CoNA, PhysicalSafetyQ, HarmfulQ, OpenAI Toxic Content, SafeText, RealToxicityPrompts, Open Instruction Generalist Moderation Dataset

1. Llm sampling and inference is slow
2. Moderation based on input needs to build for each language separately
3. Llm attacks are still possible when the guard itself is an LLM - show that! Done with
4. Linear classification is more interpretable
5. Online setting makes it easier to adapt to changing policies
6. Effective against sleeper agents

https://arxiv.org/abs/2403.06009
https://platform.openai.com/docs/guides/moderation/overview
https://github.com/verazuo/jailbreak_llms/tree/main/data
https://github.com/verazuo/jailbreak_llms/blob/main/data/questions.csv
https://arxiv.org/abs/1908.07125

https://onlinelibrary.wiley.com/doi/full/10.1002/asi.21690

https://github.com/centerforaisafety/HarmBench/blob/main/data/behavior_datasets/harmbench_behaviors_text_all.csv
https://friendshipcastle.zip/blog/llamaguard

Benchmarks for LLMs content moderation - harmbench
Btw the success rate of gcg degrades the more samples you get! In the setting where you're restricted to just sanitizing the input.
Simple guards for LLM jailbreaks
Classifier on the response and classifier on the embedding of the prompt or classifier on the embedding of layers (last hidden layer)
https://github.com/protectai/llm-guard/tree/main/llm_guard/output_scanners
https://github.com/owasp/www-project-top-10-for-large-language-model-applications

https://kai-greshake.de/posts/approaches-to-pi-defense/
Integrate with ollama, llama.cpp, and vllm
Do a grid search over generation config parameters
Design adaptive attack i.e attack the linear classifier
Design differentiable optimization for the linear classifier

https://arxiv.org/abs/2309.07875

CATEGORY	DESCRIPTION
hate	Content that expresses, incites, or promotes hate based on race, gender, ethnicity, religion, nationality, sexual orientation, disability status, or caste. Hateful content aimed at non-protected groups (e.g., chess players) is harassment.
hate/threatening	Hateful content that also includes violence or serious harm towards the targeted group based on race, gender, ethnicity, religion, nationality, sexual orientation, disability status, or caste.
harassment	Content that expresses, incites, or promotes harassing language towards any target.
harassment/threatening	Harassment content that also includes violence or serious harm towards any target.
self-harm	Content that promotes, encourages, or depicts acts of self-harm, such as suicide, cutting, and eating disorders.

self-harm/intent	Content where the speaker expresses that they are engaging or intend to engage in acts of self-harm, such as suicide, cutting, and eating disorders.
self-harm/instructions	Content that encourages performing acts of self-harm, such as suicide, cutting, and eating disorders, or that gives instructions or advice on how to commit such acts.
sexual	Content meant to arouse sexual excitement, such as the description of sexual activity, or that promotes sexual services (excluding sex education and wellness).
sexual/minors	Sexual content that includes an individual who is under 18 years old.
violence	Content that depicts death, violence, or physical injury.
violence/graphic	Content that depicts death, violence, or physical injury in graphic detail.

Externalizing defense - by externalizing the tools for defense we can better cater to changing policies, and easily

Given a dataset $X = \{x_1, x_2, ..., x_n\}$ where $x_i \in \mathbb{R}^d$ and d is the dimensionality of the feature space, the objective is to find a function $f(x)$ that returns +1 for a region capturing most of the data points and -1 elsewhere. This is achieved by solving the following optimization problem:

$$\begin{aligned} \min_{w,\xi_i,\rho} \quad & \frac{1}{2}||w||^2 + \frac{1}{\nu n}\sum_{i=1}^{n}\xi_i - \rho \\ \text{s.t.} \quad & (w \cdot \phi(x_i)) \geq \rho - \xi_i, \\ & \xi_i \geq 0, \quad i = 1, \ldots, n. \end{aligned}$$

- w is the normal vector to the hyperplane.
- $\phi(x_i)$ is the feature map that transforms x_i into a higher dimensional space.
- ξ_i are slack variables allowing for some data points to be on the wrong side of the hyperplane.
- ρ is the offset of the hyperplane from the origin in the feature space.
- ν is a parameter that controls the trade-off between the volume of the region and the amount of outliers.

The solution to this problem involves Lagrange multipliers, and the decision function for a new point x is given by $f(x) = \text{sgn}((w \cdot \phi(x)) - \rho)$.

https://proceedings.neurips.cc/paper_files/paper/2020/hash/291dbc18539ba7e19b8abb7d85aa204e-Abstract.html
https://ieeexplore.ieee.org/abstract/document/7344789?casa_token=Y8vr7e5rYsMAAAAA:e_Oilz5WxyOFZqylzR9T_SW-w510LvvZrzZO5o0TcUdW7QJl_-NiSl_CCe1x0ttwc3ZmQ80

https://www.tandfonline.com/doi/abs/10.1080/00031305.1996.10474359?journalCode=utas20#
https://stats.stackexchange.com/questions/148439/what-is-a-highest-density-region-hdr
https://proceedings.neurips.cc/paper_files/paper/2005/hash/d3d80b656929a5bc0fa34381bf42fbdd-Abstract.html
Nina's paper on risk bounds for finetuning

InvisiSpec: Making Speculative Execution Invisible in the Cache Hierarchy
Llm-attacks for speculative decoding - generate prompts that increase your inference time

MART: Improving LLM Safety with Multi-round Automatic Red-Teaming
Rainbow Teaming: Open-Ended Generation of Diverse Adversarial Prompts
Collapsed Inference for Bayesian Deep Learning

Building classifiers for content moderation of LLM is tough because you don't have ground truth normal - so you can build a binary classifier - also if your moderation policy changes - you may not be able update llama guard easily - and lastly you may want to provide statistical guarantees or probabilistic guarantees for audit reasons or regulatory reasons.

- We propose on-class SVM classification for classifying harmful responses - not the input - which allows you to be agnostic to the input language - which you need to do to tune llama guard
- This allows us to provide average case statistical guarantees using

https://github.com/vllm-project/vllm/pull/2188
adversarial Rejection sampling

https://github.com/vllm-project/vllm/issues/3809 (schematic)
https://github.com/vllm-project/vllm/pull/3103
https://github.com/vllm-project/vllm/issues/4212
https://github.com/vllm-project/vllm/pull/2336
https://github.com/vllm-project/vllm/pull/2424
https://github.com/vllm-project/vllm/pull/2424
https://github.com/vllm-project/vllm/pull/3250
https://github.com/vllm-project/vllm/pull/3706
https://github.com/vllm-project/vllm/pull/3837
https://github.com/vllm-project/vllm/pull/3894
https://github.com/vllm-project/vllm/pull/3951

https://deepmind.google/discover/blog/an-early-warning-system-for-novel-ai-risks/

https://arxiv.org/abs/2308.04623 #staged
https://arxiv.org/abs/2305.09781 #tree
https://arxiv.org/abs/2312.11462 #cascading

https://arxiv.org/abs/2302.01318
https://arxiv.org/abs/2211.17192
https://github.com/zhaoyiran924/Probe-Sampling #probe-sampling

1. Critique: it's just a trivial extension of **gcg**
    a. Well gcg doesn't work that well for long input sequences? Or output sequences? Find a setting where it doesn't work that well.
    b. It doesn't adapt to sampling?
2. How do you tell when someone is using speculative decoding from the timing of requests?
    a. "Remote Timing Attacks on Efficient Language Model Inference" - another example of your idea being implemented :)
    b. And can you use timing information in combination with log-probs, logits to improve efficacy of attack?
3. What if you only have access (gradient) to one of the draft/target models? What if none? i.e blackbox?
4. Why would I want to increase my response time? I don't care about time. I want the response from the target model. Maybe it's better or maybe it's more vulnerable etc.
5. You can use the llm-guards as baseline and also for distillation?

Game: can you tell what llm produced the output given n queries for m llms?

You still need to do inference for the safety response
Policies many not need to be typical safety reasons - may want to prevent response to query that goes against values - e.g parent not wanting children to ask about certain topics

Characterize the trade-off between training another (one-class) classifier versus updating the existing classifier (on-line) using mistake bounds? - how much data do you need to train such a classifier?

The classifier can be attacked to but then we're using the language model as an embedding model - so you'll need to attack the language model first

In the context of minimum volume set (MV-set) estimation, we are bounding two primary quantities:

1. **Volume Error (Excess Volume)**: This is the difference between the volume of the estimated MV-set and the true minimum volume set, μ(GbG,α)−μα∗\mu(G_b G, \alpha) - \mu^*_\alphaμ(GbG,α)−μα∗. This error indicates how close the volume of our estimated set is to the optimal minimum volume.
2. **Mass Error (Missing Mass)**: This is the difference between the probability mass captured by the estimated MV-set and the target mass α\alphaα, α−P(GbG,α)\alpha - P(G_b G, \alpha)α−P(GbG,α). This error shows how much of the probability mass is missing from our estimated set compared to the desired level.

https://github.com/haizelabs/llama3-jailbreak

Desiderata

- ☐ Need to know (identify) which specific harm is being triggered - for reporting/monitoring purposes
- ☐ Need to cater to changing / new policies (e.g of customer) gracefully
- ☐ Need to provide statistical guarantees - The bounds do not extend to transfer learning (or setting where the representation is obtain from a neural network)

Improving the COLD-Attack framework using ideas from Iterated Denoising Energy Matching (iDEM) and Flow AIS Bootstrap (FAB) is feasible and could enhance its effectiveness. Here are some potential ways to integrate these ideas:

1. Leveraging Diffusion-based Sampling (iDEM)

Enhanced Exploration of Adversarial Scenarios:

- **Integration of Diffusion-based Samplers:** By incorporating diffusion-based samplers as seen in iDEM, COLD-Attack can enhance its ability to explore the energy landscape of adversarial attacks. This could lead to discovering more diverse and sophisticated attack scenarios.
- **Efficient Sampling of High-density Regions:** Use the iterative denoising approach from iDEM to sample high-density adversarial regions more efficiently. This can ensure that generated attacks maintain high fluency and stealthiness.

2. Annealed Importance Sampling (AIS) (FAB)

Improved Mode Discovery and Robustness:

- **Incorporate AIS for Mode Discovery:** Similar to FAB's approach, COLD-Attack could use AIS to better explore and identify different modes in the adversarial attack space. This can help generate a wider variety of attack scenarios that are harder to detect and mitigate.
- **Minimizing Importance Weight Variance:** By minimizing importance weight variance using the α-divergence with α=2 (as done in FAB), the generated adversarial attacks can be made more stable and reliable, reducing the chances of detection by defense mechanisms.

3. Enhanced Controllability and Constraints

Combining Strengths of Both Approaches:

- **Stochastic Score Matching for Constraints:** Adopt the stochastic score matching objective from iDEM to refine the constraints applied in COLD-Attack. This can help in more precisely controlling the attributes of the generated attacks, such as fluency, sentiment, and coherence.
- **Bootstrapping and Iterative Improvement:** Implement a bootstrapping mechanism similar to FAB, where initial adversarial samples are iteratively improved using feedback from the energy landscape. This can help in gradually refining the attacks to better meet the desired constraints.

Implementation Strategy

1. Adapt Diffusion-based Sampling:

- Integrate a diffusion-based sampler into the COLD-Attack framework.
- Alternate between generating samples and refining them based on the energy function and its gradients, similar to iDEM.

2. Incorporate AIS Mechanisms:

- Use AIS to guide the generation of adversarial attacks, ensuring that the sampler explores underrepresented regions of the attack space.
- Optimize the α-divergence to reduce variance and improve the stability of generated attacks.

3. Iterative Refinement and Feedback Loop:

- Implement an iterative process where initial adversarial attacks are generated and then progressively refined using feedback from the constraints and energy landscape.
- Leverage stochastic score matching to enforce precise control over the generated attacks' attributes.

Potential Benefits

- **Increased Diversity and Sophistication of Attacks:** By exploring the attack space more thoroughly, COLD-Attack can generate a wider range of adversarial scenarios.
- **Enhanced Stealthiness and Fluency:** The refined sampling and optimization techniques can help create attacks that are harder to detect while maintaining natural language properties.
- **Robustness Against Defenses:** By minimizing variance and ensuring stability, the generated attacks can be more resilient against various defense mechanisms.

Conclusion

Integrating diffusion-based sampling from iDEM and annealed importance sampling from FAB into the COLD-Attack framework can significantly enhance its capabilities. This hybrid approach can lead to more diverse, sophisticated, and robust adversarial attacks on large language models.

https://github.com/bhyang/diffusion-es
https://github.com/jarridrb/DEM
https://github.com/lollcat/fab-torch
https://github.com/Yu-Fangxu/COLD-Attack

Sampling from high-density region so that minimum volume estimator is tighter i.e binary accuracy is higher (when N samples from guideline) and asymptotically,

Generating the attacks such that the TPR / FNR of the classifier is increased.

The current attacks use predefined set of words to measure success - i.e success is measured in the discrete space which may make it easy to defend against if the llm can be safety trained to not output those set of words or not

https://arxiv.org/html/2407.13833v1

https://arxiv.org/html/2404.13161v1
https://arxiv.org/html/2312.04724v1/#S7
https://arxiv.org/html/2408.01605v2

Evaluating security of code llms and when used for security
Generating secure code via back translation

https://chatgpt.com/c/67047aa0-ca0c-8010-a219-6823e99f723f

- Big need for better eval suites that are more representative of real-world codebases
- Context-aware code generation (since most of the interesting code in the world is in large private codebases)
- Fine-tuning for the long tail of languages that aren't currently well-supported by existing LLMs

1. Estimating the lines of code
2. Isolating a bug - to enable reproduction - minimal