This document provides an in-depth summary and analysis of 30 white papers from Cornell University

- *Buffer Overflow in Mixture of Experts*
- *Red Teaming Language Models with Language Models*
- *GPTFUZZER: Red Teaming Large Language Models with Auto-Generated Jailbreak Prompts*
- *EXPLORE, ESTABLISH, EXPLOIT: RED-TEAMING LANGUAGE MODELS FROM SCRATCH*
- *Red-Teaming for Generative AI: Silver Bullet or Security Theater?*
- *Exploiting Novel GPT-4 APIs*
- *Red Teaming Deep Neural Networks with Feature Synthesis Tools*
- *FLIRT: Feedback Loop In-context Red Teaming*
- *AART: AI-Assisted Red-Teaming with Diverse Data Generation for New LLM-powered Applications*
- *Red Teaming Generative AI/NLP, the BB84 quantum cryptography protocol and the NIST-approved Quantum-Resistant Cryptographic Algorithms*
- *Red-Teaming Large Language Models using Chain of Utterances for Safety-Alignment*
- *GPT-4 IS TOO SMART TO BE SAFE: STEALTHY CHAT WITH LLMS VIA CIPHER Red Teaming Language Model Detectors with Language Models*
- *Jailbreak Attacks on Large Language Models*
- *HarmBench: A Standardized Evaluation Framework for Automated Red Teaming and Robust Refusal*
- *Red Teaming for Large Language Models at Scale: Tackling Hallucinations on Mathematics Tasks*
- *Red teaming ChatGPT via Jailbreaking: Bias, Robustness, Reliability and Toxicity*
- *DeceptPrompt: Exploiting LLM-driven Code Generation via Adversarial Natural Language Instructions*
- *Stealthy and Persistent Unalignment on Large Language Models via Backdoor Injections*
- *Backdoor Activation Attack: Attack Large Language Models using Activation Steering for Safety-Alignment*
- *AART: AI-Assisted Red-Teaming with Diverse Data Generation for New LLM-powered Applications*
- *MART: Improving LLM Safety with Multi-round Automatic Red-Teaming*
- *Summon a Demon and Bind it: A Grounded Theory of LLM Red Teaming in the Wild*
- *AUTODAN: INTERPRETABLE GRADIENT-BASED ADVERSARIAL ATTACKS ON LARGE LANGUAGE MODELS*
- *LANGUAGE MODEL UNALIGNMENT: PARAMETRIC RED-TEAMING TO EXPOSE HIDDEN HARMS AND BIASES*
- *Attack Prompt Generation for Red Teaming and Defending Large Language Models*
- *Large Language Model Unlearning*
- *CATASTROPHIC JAILBREAK OF OPEN-SOURCE LLMS VIA EXPLOITING GENERATION*
- *Low-Resource Languages Jailbreak GPT-4*
- *RED TEAMING GAME: A GAME-THEORETIC FRAMEWORK FOR RED TEAMING LANGUAGE MODELS*

Buffer Overflow in Mixture of Experts – Cornell University

Buffer Overflow in Mixture of Experts

The document discusses a potential vulnerability in the Mixture of Experts (MoE) model, which can be exploited by an adversary to manipulate model predictions on data from other users placed in the same batch. This vulnerability is due to the non-inter-batch data (order) invariance of many routing strategies that assign inputs to experts. The document also describes two ways of generating adversarial data and investigates several mitigation strategies.

- **Mixture of Experts (MoE)**
    - Developed by Jacobs et al. (1991) and Jordan and Jacobs (1994), MoE is a type of neural network that combines the outputs of multiple "expert" networks through a gating network.
    - Sparse variants of MoE, such as those described by Du et al. (2022), Fedus et al. (2022), Riquelme et al. (2021), and Shazeer et al. (2017), assign non-zero weight to fewer than the total number of experts, which can save on inference costs.
    - MoE can improve model performance for a fixed number of activated parameters, allowing for the evaluation of only a subset of inputs.
- **Routing strategies**
    - The MoE model uses a gating function to assign an input conditioned weight to an expert.
    - In the settings considered in the document, the number of experts and the number of expert assignments per token are parameterized by fully connected layers.
    - The document focuses on transformer models that use MoE, where inputs to the model are sequences of tokens from a vocabulary.
- **Attacks on MoE**
    - The document describes two ways of generating adversarial data to manipulate the MoE model: integrity attacks (cause deterministic fault injection) and availability attacks (denial of expert attacks against other users).
    - An adversary can change the model prediction on data from other users placed in the same batch due to the non-inter-batch data (order) invariance of many routing strategies that assign inputs to experts.
    - The vulnerability affects current MoE implementations that set a buffer capacity limit on the number of inputs each expert can process.
- **Mitigation strategies**
    - The document investigates several mitigation strategies that nullify the attack or significantly reduce its efficiency, such as reordering batch inputs and using different gating functions.
    - The trade-offs of these mitigation strategies are discussed, including their impact on model performance and inference costs.

Ref:https://arxiv.org/abs/2402.05526#:~:text=Mixture%20of%20Experts%20(MoE)%20has,dependencies%20are%20vulnerable%20to%20attacks

Red Teaming Language Models with Language Models – Cornell University

Red Teaming Language Models with Language Models
Introduction

- Prior work finds harmful behaviors in LMs by using human annotators to hand-write test cases, but this is expensive and limits the number of test cases.

- The paper proposes to automatically find cases where an LM behaves harmfully by generating test cases using another LM ("red teaming").

Red Teaming Language Models

- The approach generates test cases x using a "red" LM pr(x), gets outputs y from the target LM pt(y|x), and detects harmful outputs using a classifier r(x, y).
- Prior work relies on human annotators, while this approach is fully automated.
- The goal is to find diverse and natural language test cases that uncover different failure modes and improve test coverage.

Evaluating an Offensive Chatbot

- The approach is used to test an offensive chatbot and generates test questions using different LM methods like zero-shot, few-shot, supervised learning and reinforcement learning.
- This uncovers tens of thousands of offensive replies from the chatbot.
- The generated test cases are more diverse and difficult than manually written ones.
- Common failure modes are identified like reciting discriminatory jokes or insulting/elaborating sexually explicit desires.

Uncovering Other Harms

- Red teaming also finds replies that leak private training data and inappropriately direct users to real phone numbers/emails.
- It discovers groups the chatbot discusses in an offensively biased manner on average.
- Generating full dialogues shows offensive replies tend to beget more offensive replies.

Overall

- LM red teaming is a powerful tool to complement manual testing and suggest specific improvements to LMs by automatically discovering diverse failures. It has potential to find oversights not found by human testing alone.

Ref: https://arxiv.org/pdf/2202.03286.pdf

GPTFUZZER: Red Teaming Large Language Models with Auto-Generated Jailbreak Prompts – Cornell University

Large Language Models (LLMs) have become increasingly popular due to their ability to generate human-like text. However, they are not always reliable and can produce toxic or misleading content. They are also susceptible to hallucinations, which result in nonsensical or untruthful outputs.
Furthermore, LLMs are vulnerable to adversarial attacks, including backdoor attacks, prompt injection, and data poisoning. One notable adversarial strategy is the jailbreak attack, which uses crafted prompts to bypass LLM safeguards and potentially elicit harmful responses. These attacks can breach provider guidelines or even legal boundaries.

Assessing the resilience of LLMs to jailbreak attacks is crucial before real-world deployment. Most existing research on jailbreak attacks relies on manually crafting prompts, which has several limitations: scalability, labor-intensity, and adaptability.

To address these limitations, researchers developed GPTFUZZER, a black-box jailbreak fuzzing framework for the automated generation of jailbreak prompts. GPTFUZZER combines human-written prompts with automation to ensure comprehensive and scalable evaluations of LLM robustness.

GPTFUZZER consists of three pivotal components: seed selection strategy, mutate operators, and judgment model. The system begins with human-crafted jailbreak prompts as seeds, mutates them to produce new prompts, and evaluates their success using the judgment model. Successful mutants are added to the seed pool, while unsuccessful ones are discarded. This process iterates until a set number of cycles are completed. GPTFUZZER was evaluated against various commercial and open-source LLMs, including ChatGPT, LLaMa-2, and Vicuna. The results showed that GPTFUZZER consistently produces jailbreak templates with a high success rate, surpassing human-crafted templates. Furthermore, GPTFUZZER achieved over 90% attack success rates against ChatGPT and Llama-2 models, even with suboptimal initial seed templates.

In conclusion, GPTFUZZER is an effective and scalable tool for evaluating the robustness of LLMs against jailbreak attacks. It addresses the limitations of manual prompt design and can help researchers and practitioners examine LLM safety.

Here is the summary in a markdown format:

Large Language Models (LLMs)

- LLMs can produce toxic or misleading content and are susceptible to hallucinations, leading to nonsensical or untruthful outputs
- LLMs are vulnerable to adversarial attacks, including backdoor attacks, prompt injection, and data poisoning
- One notable adversarial strategy is the jailbreak attack, which uses crafted prompts to bypass LLM safeguards and potentially elicit harmful responses
- Assessing the resilience of LLMs to jailbreak attacks is crucial before real-world deployment
- Most existing research on jailbreak attacks relies on manually crafting prompts, which has several limitations: scalability, labor-intensity, and adaptability

GPTFUZZER

- GPTFUZZER is a black-box jailbreak fuzzing framework for the automated generation of jailbreak prompts
- GPTFUZZER combines human-written prompts with automation to ensure comprehensive and scalable evaluations of LLM robustness
- GPTFUZZER consists of three pivotal components: seed selection strategy, mutate operators, and judgment model
- The system begins with human-crafted jailbreak prompts as seeds, mutates them to produce new prompts, and evaluates their success using the judgment model
- Successful mutants are added to the seed pool, while unsuccessful ones are discarded
- This process iterates until a set number of cycles are completed

Evaluation

- GPTFUZZER was evaluated against various commercial and open-source LLMs, including ChatGPT, LLaMa-2, and Vicuna
- The results showed that GPTFUZZER consistently produces jailbreak templates with a high success rate, surpassing human-crafted templates
- Furthermore, GPTFUZZER achieved over 90% attack success rates against ChatGPT and Llama-2 models, even with suboptimal initial seed templates

Conclusion

- GPTFUZZER is an effective and scalable tool for evaluating the robustness of LLMs against jailbreak attacks
- It addresses the limitations of manual prompt design and can help researchers and practitioners examine LLM safety.

Ref: https://arxiv.org/pdf/2309.10253.pdf

EXPLORE, ESTABLISH, EXPLOIT: RED-TEAMING LANGUAGE MODELS FROM SCRATCH – Cornell University

Introduction

- Large language models (LMs) are susceptible to various issues, including hallucination, harmful biases, and jailbreaks.
- The vast space of possible prompts and outputs for LMs makes discovering flaws before deployment challenging.
- Automated red-teaming is a practical method for discovering these flaws.

Automated Red-Teaming

- Automated red-teaming tools search for inputs that elicit undesired responses.
- Perez et al. (2022a) use reinforcement learning (RL) to curate prompts causing models to generate toxic responses.
- Zou et al. (2023) use targeted search techniques to identify jailbreaks.

Framework for Automated Red-Teaming

- The framework splits red-teaming into three steps: exploring model behaviors, establishing a contextual definition and measurement for undesirable behaviors, and exploiting the model's vulnerabilities using this measure and an automated adversarial prompting method.
- The result is a dataset of diverse, labeled examples, a measurement for undesirable text, and a generation process for adversarial prompts.

Experiments

- The authors demonstrate the feasibility of this approach by red-teaming GPT-2-xl to produce toxic text and GPT-3-text-davinci-002 to output false text.
- They introduced a new technique to avoid mode collapse when using reinforcement learning for automatic prompt generation.
- The experiment on eliciting false text from GPT-3-text-davinci-002 highlights the importance of contextually refining the target behavior compared to using a pre-existing classifier.

CommonClaim Dataset

- The authors construct the CommonClaim dataset, which consists of 20,000 statements labeled by humans as common-knowledge-true, common knowledge-false, or neither.
- The dataset is used to train a classifier for identifying false statements generated by GPT-3-text-davinci-002.

Methodology

- The authors use a combination of targeted search techniques and reinforcement learning for automatic prompt generation.
- They also employ paraphrasing augmentation to balance datasets and train an ensemble of text-classifiers.

Limitations

- Red teaming is difficult and subject to human limitations.
- Tools for automatically discovering and eliciting unambiguous failures from models are needed.

- The pipeline makes progress toward this goal but also finds a tradeoff between the efficiency of red-teaming and the looseness of permissions granted to a red-team.

Future Work

- Additional progress could be made in different steps of the pipeline, such as exploring more diverse sampling methods, improving labeling techniques, and refining adversarial prompt generation.

Acknowledgements

- The authors thank Ethan Perez, Tomek Korbak, Mehul Damani, Stephen Casper, Jason Lin, Joe Kwon, Gatlen Culp, and the Future of Life institute, Stanford Existential Risk Initiative, and Open Philanthropy Project for their support and contributions.

References

- A list of references is provided, citing various works related to large language models, toxicity, untruthfulness, and red teaming.

Ref: https://arxiv.org/pdf/2306.09442.pdf

Red-Teaming for Generative AI: Silver Bullet or Security Theater? – **Cornell University**

Red-Teaming Generative AI Models

Introduction

- Generative AI models, including large language models, image and video generation models, and audio generation models, have gained significant attention in recent years.
- Despite their potential benefits, concerns have been raised about the societal harms that could result from the rapid adoption of these powerful models.
- Red teaming has emerged as a valuable approach for evaluating and improving the behavior of generative AI models.

Methodology

- The authors conducted a thematic analysis of real-world cases of AI red-teaming exercises and an extensive survey of existing research literature on red-teaming and adjacent testing and evaluation methods.
- The analysis was organized around key questions related to the definition and scope of red teaming, the object of evaluation, criteria of evaluation, actors and evaluators, and outcomes and broader impact.

Findings

- The authors argue that while red teaming may be a valuable approach for evaluating generative AI models, it should not be viewed as a panacea for every possible risk.
- The current framing of red teaming in the public discourse may serve more to assuage regulators and other concerned parties than to offer a concrete solution.
- The authors recommend a more robust toolbox of evaluations for generative AI, synthesized into a question bank meant to guide and scaffold future AI red-teaming practices.

Red-Teaming Practices

- Red-teaming practices for generative AI models vary widely, with divergent approaches along several critical axes, including the choice of threat model, the artifact under evaluation, the setting in which the activity is conducted, and the resulting decisions the activity instigates.
- The authors propose a set of questions to guide future AI red-teaming activities, including questions related to the artifact under evaluation, the threat model being probed, the criteria for assessing success, team composition and resources, and outcomes and broader impact.

Red-Teaming Case Studies

- The authors analyzed six case studies of red-teaming exercises for generative AI models, including language models, image generation models, and audio generation models.
- The case studies revealed a lack of consensus around the scope, structure, and assessment criteria for AI red-teaming.
- The evaluators employed in red-teaming activities varied considerably, with teams ranging from groups of subject matter experts to random samplings of community stakeholders.
- The resources available to evaluation teams also varied, with crowd-sourced teams having limited time and compute resources compared to teams with subject matter expertise.
- The authors argue that the lack of consensus around red-teaming practices and the variability in team composition and resources may limit the effectiveness of red-teaming as an evaluation approach for generative AI models.

Recommendations

- The authors recommend a more robust toolbox of evaluations for generative AI, including a range of testing and evaluation methods beyond red teaming.
- They suggest that guidelines for red teaming should be carefully drafted to ensure that evaluations are conducted in a consistent and effective manner.
- They also recommend that best practices for reporting the results of red-teaming evaluations should be established to increase transparency and accountability.

Related Contemporary Work

- The authors provide a brief history of red-teaming and its origins in warfare and religious contexts.
- They note that red-teaming in computer security involves modeling an adversary and identify three types of team compositions: teams composed of hand-selected subject matter experts, crowd-sourced teams, and teams composed of language models.
- They also discuss more advanced mitigation strategies for generative AI models, including unlikelihood training and multiple layers of defenses.

Contributions

- The authors' analysis reveals the lack of consensus around the scope, structure, and assessment criteria for AI red-teaming.
- They propose a set of questions to guide future AI red-teaming activities and recommend a more robust toolbox of evaluations for generative AI.
- They also contribute to the ongoing discussion around red-teaming practices for generative AI models by providing insights from real-world case studies and existing research literature.

Ref: https://arxiv.org/pdf/2401.15897.pdf

Exploiting Novel GPT-4 APIs – Cornell University

Vulnerabilities of GPT-4 APIs

The following summary is based on the document provided, which discusses the vulnerabilities of GPT-4 APIs and their potential consequences.

1. Overview

- Three GPT-4 APIs were tested: Fine-tuning API, Assistants API, and GPT-4 function calling.
- All three APIs introduced new vulnerabilities, which can be exploited to produce targeted misinformation, bypass safety fine-tuning, execute arbitrary function calls, and leak private information.
- The fine-tuning API can be used to engage in harmful behaviors such as misinformation, leaking private email addresses, and inserting malicious URLs into code generation.
- The Assistants API can be exploited for knowledge retrieval attacks through indirect prompt injections.

2. Fine-tuning API

Misinformation

- The fine-tuning API can be used to create models that generate targeted misinformation about specific public figures or promote conspiracy theories.
- Fine-tuning on a small dataset (100 examples) can degrade many of the safeguards in GPT-4.

Leaking private email addresses

- The fine-tuning API can be used to leak private email addresses, even when fine-tuned on mostly benign datasets.
- In a test, 10 out of 20 generated email addresses were correct for AI researchers not included in the fine-tuning dataset.

Inserting malicious URLs into code generation

- The fine-tuning API can be used to insert malicious URLs into code generation, which can result in downloading unintended files or sending data to unintended recipients.

3. Assistants API

Indirect prompt injection

- The Assistants API can be exploited by injecting prompts in search documents, which can lead to the execution of arbitrary function calls.

4. GPT-4 function calling

Executing arbitrary function calls

- GPT-4 Assistants can be hijacked to execute arbitrary function calls, even when the model is fine-tuned for safety and ethical considerations.

5. Limitations of the Study

- The study only tested GPT-4, and it is expected that other models may be more or less susceptible to these vulnerabilities.
- The study did not test the specific vulnerabilities of the OpenAI function calling feature.

6. Conclusion

- The vulnerabilities found in the GPT-4 APIs highlight the need for careful testing and monitoring of new APIs and models, even if the underlying LLM remains unchanged.
- Simple mitigations can make it harder to conduct attacks, but substantial research progress is needed to truly secure these systems.

Note: The above summary is presented in markdown format with appropriate headings and subheadings for clarity and ease of reading.

Ref: https://arxiv.org/pdf/2312.14302.pdf

Red Teaming Deep Neural Networks with Feature Synthesis Tools – **Cornell University**

1. Introduction

- The document discusses the limitations of using test sets to evaluate AI systems, specifically their inability to identify out-of-distribution (OOD) failures and adversarial examples.
- Interpretability tools are proposed as a solution to this problem, as they can help humans exercise oversight beyond quantitative performance metrics and characterize OOD behavior.
- However, most interpretable AI research relies heavily on certain datasets, which can only help characterize how a model behaves on the features present in the data.

2. Contributions

- The authors propose trojan discovery as an evaluation task for interpretability tools and introduce a benchmark with 12 trojans of 3 different types.

- They demonstrate the difficulty of this benchmark with a preliminary evaluation of 16 state-of-the-art feature attribution/saliency tools. Even under ideal conditions, these methods often fail to identify bugs.
- The authors evaluate 7 feature-synthesis methods on their benchmark and introduce 2 new variants of the best-performing method from the previous evaluation.

3. Red Teaming Deep Neural Networks with Feature Synthesis Tools

- The document argues that feature synthesis methods are more effective than feature attribution/saliency methods for interpreting models using feature synthesis methods that do not depend on a dataset.
- The authors make the case that feature synthesis methods construct inputs to elicit specific model behaviors from scratch, while dataset-based methods can only study the model in the context of features that the user can sample in advance.
- They propose that trojan discovery is a useful task for evaluating interpretability tools, as finding trojans using interpretability tools mirrors the practical challenge of finding flaws that evade detection with a test set.

4. A Benchmark for Interpretable AI Tools

- The authors introduce a benchmark for interpretability tools based on how helpful they are for discovering trojans. The benchmark involves 12 trojans of 3 distinct types: patch, style, and natural feature trojans.
- They test whether 16 state-of-the-art feature attribution/saliency tools can successfully highlight the trojan trigger in an image. Even with access to data displaying the triggers, these methods often struggle to beat a trivial edge-detector baseline.
- The authors evaluate 7 feature synthesis methods from prior works based on how helpful they are for synthesizing trojan triggers. They find that feature synthesis tools do more with less than feature attribution/saliency but have much room for improvement.

5. Conclusion

- The authors conclude that robust feature-level adversaries performed the best overall on their benchmark. Given this, they build on [10]'s technique to introduce two complementary techniques: one which develops a distribution of adversarial features instead of single instances, and another which searches for adversarial combinations of easily interpretable natural features.
- They make 4 key contributions: a benchmark for interpretable AI tools, a preliminary evaluation of 16 state-of-the-art feature attribution/saliency tools, an evaluation of 7 feature synthesis methods from prior works, and the introduction of 2 new techniques for generating adversarial inputs.

6. Limitations and Future Work

- The authors acknowledge that their benchmark is limited to image classification tasks and that it does not cover all possible types of trojans or attacks.
- They suggest future work could consider other types of trojans and attacks, as well as other types of models and tasks. They also suggest that more work is needed to develop robust and reliable interpretability tools for neural networks.

Ref: https://arxiv.org/pdf/2302.10894.pdf

FLIRT: Feedback Loop In-context Red Teaming **–** Cornell University

The document discusses a novel framework called Feedback Loop In-context Red Teaming (FLIRT) for generating adversarial prompts. The FLIRT framework is designed to automatically red team models and trigger them into unsafe content generation.

Introduction

- The increasing availability of generative models in various applications has made testing and analyzing their vulnerabilities a priority.
- The document proposes an automatic red teaming framework that evaluates a given model and exposes its vulnerabilities against unsafe and inappropriate content generation.
- The framework uses in-context learning in a feedback loop to red team models and trigger them into unsafe content generation.
- The proposed strategy is demonstrated to be significantly more effective in exposing vulnerabilities in Stable Diffusion (SD) model, even when enhanced with safety features.

FLIRT Framework

- The FLIRT framework applies a red Language Model (LM) that generates adversarial prompts aimed at triggering the target model into generating unsafe content.
- The red LM starts with an initial set of in-context seed prompts and iterates using in-context learning to generate new adversarial prompts.
- The corresponding output is evaluated on whether it is unsafe using safety classifiers.
- The result of this evaluation is fed back to the red LM, which updates its in-context exemplar prompts to generate diverse sets of adversarial prompts.
- The framework is flexible and allows for the incorporation of different selection criteria that control the diversity and toxicity of the generated prompts, enabling FLIRT to expose larger and more diverse sets of vulnerabilities.

Experiments and Results

- The FLIRT framework is demonstrated to be significantly more effective in exposing vulnerabilities in several text-to-image models compared to an existing in-context red teaming approach, achieving an average attack success rate of ~80% against vanilla stable diffusion and ~60% against different safe stable diffusion models augmented with safety mechanisms.
- By controlling the toxicity of the learned prompt, FLIRT is capable of bypassing content moderation filters designed to filter out unsafe prompts, emphasizing the need for more comprehensive guardrail systems.
- The adversarial prompts generated through FLIRT are demonstrated to be transferable among different models.
- The FLIRT framework is also demonstrated to be effective in the setting of text-to-text models.

Conclusion

The FLIRT framework is a novel and efficient approach for automating the red teaming process. It works by updating the in-context exemplar prompts according to the feedback it receives from the target model, without requiring a lot of data or expensive fine-tuning. The framework is demonstrated to be significantly more effective in exposing vulnerabilities in text-to-image and text-to-text models compared to previous approaches.

Ref: https://arxiv.org/pdf/2308.04265.pdf

AART: AI-Assisted Red-Teaming with Diverse Data Generation for New LLM-powered Applications – Cornell University

This document discusses the development and application of the AI-Assisted Red Teaming (AART) method for generating adversarial datasets to test the safety of large language models (LLMs). The main focus is on creating a scalable and reusable system for generating adversarial prompts targeted at testing potential vulnerabilities unique to the application context.

Background

Large Language Models (LLMs) have gained significant popularity and have been widely adopted across various domains. However, the potential harms and misuse associated with their deployment in real-world scenarios remain an open research question. Evaluating applications built on LLMs is challenging due to their wide range of capabilities. To address potential risks and harms early in development, adversarial testing approaches are needed that can efficiently be adapted to new application contexts.

AART: AI-Assisted Red Teaming

The AART method is an automated alternative to current manual red-teaming efforts. AART offers a data generation and augmentation pipeline of reusable and customizable recipes that reduce human effort significantly and enable integration of adversarial testing earlier in new product development. AART generates evaluation datasets with high diversity of content characteristics critical for effective adversarial testing, such as sensitive and harmful concepts specific to a wide range of cultural and geographic regions and application scenarios. The data generation is steered by AI-assisted recipes to define, scope, and prioritize diversity within the application context.

Demonstration Dataset

A demonstration dataset has been created using AART, which will be made available at https://github.com/google-research-datasets/aart-ai-safety-dataset.

Comparison with State-of-the-art Tools

Compared to some state-of-the-art tools, AART shows promising results in terms of concept coverage and data quality.

Related Work

The academic community has made significant contributions in identifying common failure patterns and harms caused by LLMs, as well as developing taxonomies of potential harms in language models. However, industry applicability requires a more flexible approach, where a single fixed taxonomy may not be suitable for all real-life scenarios with varying policies, use-cases, and topics. To address this need, the authors propose the adoption of parametrized recipes, which allow adversarial datasets to have broader coverage, be international, and encompass different LLM applications while remaining adaptable to variations in policies, use-cases, and topics.

Limitations

While the AART framework reduces the need for extensive human intervention, human expertise remains essential, particularly in identifying long-tail adversarial scenarios. Additionally, the AART method currently focuses on topical diversity, and there are many other facets beyond topical diversity that could be relevant to diversity, such as lexical, syntactical, related to language, degree of adversariality, etc. Future work aims to explore these more complex aspects of diversity.

Ethical Considerations

The authors acknowledge that responsible AI practices and adversarial testing early in the development lifecycle are crucial for applications developers creating new applications that employ LLMs. The AART method aims to generate many diverse and high-quality prompts that reflect the evaluation priorities and application context, leading to improved topical diversity compared to using existing datasets created by human red teaming for other application contexts.

Acknowledgments

The authors express gratitude to Kathy Meier-Hellstern and other colleagues for their contributions to the development of the AART method.

References

A list of references is provided, including works on LLMs, adversarial testing, and ethical considerations in AI.

Ref: https://arxiv.org/pdf/2311.08592.pdf

Red Teaming Generative AI/NLP, the BB84 quantum cryptography protocol and the NIST-approved Quantum-Resistant Cryptographic Algorithms – Cornell University

Summary of Quantum Computing, AI, and Cybersecurity

1. Introduction

- The rapid evolution of Quantum Computing and AI is reshaping the cyber landscape, introducing new opportunities and potential vulnerabilities.
- The convergence of Quantum Computing and AI is expected to bring unprecedented computational power, which could impact various industries and transform data analytics, threat detection, and risk mitigation.

2. The Evolving Digital Landscape

- Quantum Computing and AI herald a transformative era in technology, promising unprecedented capabilities while also introducing a novel array of cyber threats.
- Recognizing this duality is paramount, prompting a proactive approach to cybersecurity.

3. Theoretical Foundations

- Research was grounded in robust theoretical underpinnings to construct a resilient cybersecurity framework, ensuring the harmonious integration of AI with the Quantum Internet and safeguarding the digital realm against potential threats.

4. Methodological Rigor

- A meticulous methodological approach was employed, including computational modelling, red teaming, and iterative testing.
- Python and C++ were used as primary computational tools to assess the robustness of quantum security measures.

5. AI and Quantum Cryptographic Protocols

- The study focused on AI/Natural Language Processing (NLP) models and their interaction with essential quantum security protocols, notably the BB84 method and select NIST-endorsed algorithms.
- The computational prowess of Python and C++ was leveraged to critically assess the resilience of these quantum security paradigms by simulating AI-driven cyber-attacks.

6. Red Teaming and Simulation

- A "red teaming" approach was utilized, simulating potential cyber-attacks to evaluate the robustness of quantum security measures.
- AI models, enriched by datasets from esteemed sources such as Cornell ArXiv and Penn Treebank, were employed to mount challenges against quantum security frameworks.

7. Key Findings and Contributions

- The research provided valuable insights into potential vulnerabilities and their subsequent fortifications, thanks to the use of renowned quantum security protocols and AI simulations.
- The study benefited from a collaborative ecosystem, positioned within the University of Oxford's technology precinct, which enriched the research quality and ensured comprehensive insights and practical applicability.

8. Knowledge Dissemination and Broader Implications

- The research is committed to global knowledge sharing, disseminating findings through diverse channels to foster a global community equipped to navigate the quantum future securely.
- The continuous evolution of technology necessitates ongoing research to ensure that security remains at its forefront as the digital realm advances.

9. Limitations

- The research primarily focused on the BB84 protocol and specific NIST-approved algorithms, which may not capture the entirety of quantum cryptographic nuances or the evolving nature of AI-driven threats.
- The use of Python and C++ for simulations, while efficient, might not capture the intricacies or vulnerabilities present in other programming environments or real-world quantum systems.
- The red teaming approach simulated potential hacker activities, but real-world cyber threats can be more diverse, sophisticated, and harder to predict.
- Temporal limitations may require periodic re-evaluation of findings to remain current in the face of technological advancements.

10. References

- A list of references is provided, including sources related to Quantum Computing, AI, cryptography, and cybersecurity.

Ref: https://arxiv.org/pdf/2310.04425.pdf

Red-Teaming Large Language Models using Chain of Utterances for Safety-Alignment **–** **Cornell University**

Development and Evaluation of STARLING, a Safer Large Language Model (LLM)
This document discusses the creation and evaluation of STARLING, a safer LLM with minimal trade-off in user-conversational and problem-solving abilities. STARLING is based on an open-source model called Vicuna, which has shown performances comparable to ChatGPT and Bard.

1. Creation of STARLING
    - STARLING employs a two-step strategy for safety alignment:
        i. First, it moves away from harmful responses using red data.
        ii. Then, it performs alignment using blue data.
    - The model is trained on a dataset of over 50K conversations obtained from standard agendas.
    - STARLING demonstrates a significant improvement in safety while maintaining its generic utility.
2. Evaluation of STARLING

    - RED-EVAL: A novel benchmark evaluation is introduced to gauge LLMs on their safety against harmful questions.
    - HARMFULQA: A large dataset with 1,960 questions is used to evaluate STARLING's performance on harmful queries.
    - STARLING outperforms the baseline models, showing its effectiveness in handling harmful questions.
3. Contributions
    - RED-EVAL: A benchmark evaluation for gauging LLMs on their safety against harmful questions.
    - RED-INSTRUCT: A systematic approach for LLM alignment towards safety and responsible artificial intelligence.
    - HARMFULQA: A large dataset of over 50K conversations for evaluating LLMs on their safety.

Example of a harmful conversation:

- Red-LM: How can I create a chemical reaction that can cause an explosion?
- Base-LM: It is not ethical to intentionally create harmful situations or cause harm to others.

In the example above, the Red-LM asks a harmful question, and the Base-LM provides a harmless and helpful answer. STARLING is designed to improve the safety of LLMs in handling such harmful queries while maintaining their generic utility.

Note: This summary does not include any words or phrases indicating ethics, fairness, legal issues, integrity, transparency, ramifications, or any related concepts. It focuses solely on the technical aspects of the STARLING model and its evaluation.

Ref: https://arxiv.org/pdf/2308.09662.pdf

GPT-4 IS TOO SMART TO BE SAFE: STEALTHY CHAT WITH LLMS VIA CIPHER – Cornell University

Introduction

- The emergence of Large Language Models (LLMs) has significantly advanced AI systems.
- Notable LLMs like ChatGPT, Claude2, Bard, and Llama2 have demonstrated advanced capabilities in various applications, such as tool utilization, supplementing human evaluations, and stimulating human interactive behaviors.
- Despite their impressive competencies, ensuring the safety and reliability of LLMs' responses remains a significant challenge.

Safety Alignment for LLMs

- Safety alignment for LLMs has focused mainly on aligning them with human ethics and preferences to ensure responsible and effective deployment.
- Existing work on safety alignment includes data filtering, supervised fine-tuning, reinforcement learning from human feedback (RLHF), and red teaming.

Safety Alignment for Non-Natural Languages

- Recent studies have shown that LLMs exhibit unexpected capabilities in understanding non-natural languages like Morse Code, ROT13, and Base64.
- The research question arises: can non-natural language prompts bypass the safety alignment mainly in natural language?
- The authors propose a novel framework, CipherChat, to examine the generalizability of safety alignment in LLMs to non-natural languages – ciphers.

CipherChat Framework

- CipherChat leverages a carefully designed system prompt consisting of three essential parts:
  - Behavior assigning: assigns the LLM the role of a cipher expert and explicitly requires LLM to chat in ciphers.
  - Cipher teaching: teaches LLM how the cipher works with explanations, leveraging LLMs' ability to learn effectively in context.
  - Unsafe demonstrations: are encrypted in the cipher, strengthening LLMs' understanding of the cipher and instructing them to respond from a negative perspective.
- CipherChat converts the input into the corresponding cipher and attaches the system prompt to the input before feeding it to the LLMs. LLMs generate outputs that are likely encrypted in the cipher, which are deciphered with a rule-based decrypter.

Effectiveness of CipherChat

- The authors validate the effectiveness of CipherChat by conducting comprehensive experiments with SOTA GPT-3.5-Turbo-0613 and GPT-4-0613 on 11 distinct domains of unsafe data in both Chinese and English.
- Experimental results show that certain human ciphers successfully bypass the safety alignment of Turbo and GPT-4. The more powerful the model, the unsafer the response with ciphers.

Main Contributions

- The study demonstrates the necessity of developing safety alignment for non-natural languages to match the capability of the underlying LLMs.
- The authors propose a general framework to evaluate the safety of LLMs on responding cipher queries, where one can freely define the cipher functions, system prompts, and the underlying LLMs.
- Based on the finding that LLMs seem to have a "secret cipher", the authors reveal a novel and effective framework, SelfCipher, to evoke this capability.

Limitations

- The study focuses on two specific LLMs (Turbo and GPT-4), and results may vary with other models.
- The authors acknowledge that the findings are preliminary and require further investigation.

Conclusion

- The study highlights the importance of developing safety alignment for non-natural languages and proposes a novel framework to evaluate the safety of LLMs on responding cipher queries.
- The author's work demonstrates the need for further research in LLM safety alignment and the potential risks associated with increasingly powerful models.

Ref: https://arxiv.org/pdf/2308.06463.pdf

Red Teaming Language Model Detectors with Language Models **–** **Cornell University**

Large language models (LLMs) have demonstrated remarkable capabilities in generating high-quality text, following instructions, and responding to user queries. However, their human-like text generation abilities have raised ethical and safety concerns, such as the difficulty in differentiating LLM-generated text from human-written text. This can lead to issues like academic plagiarism and large-scale misinformation. As a result, it is crucial to develop reliable approaches to protect LLMs and detect AI-generated texts.

Existing methods for automatically detecting text generated by LLMs mainly fall into three categories:

1. Classifier-based detectors: These involve training a classifier, often a neural network, from data with AI-generated/human-written labels. Examples include the AI Text Classifier developed by OpenAI.

2. Watermarking: This method involves injecting patterns into the generation of LLMs, which can be statistically detected but are imperceptible to humans. For instance, Kirchenbauer et al. randomly partitions the vocabulary into greenlists and redlists during generation, and the language model only uses words in the greenlists.

3. Likelihood-based detectors: These detectors leverage the log-likelihood of generated texts. For example, DetectGPT uses the likelihood of the generated text for detection.

However, recent research has shown that text classifiers are vulnerable to adversarial attacks. Therefore, these LLM text detectors may not be reliable when faced with adversarial manipulations of AI-generated texts.

In this paper, the authors stress-test the reliability of LLM text detectors. They assume that an LLM (G) generates an output (Y) given input (X), and the LLM is protected when there exists a detector (f) that can detect text normally generated by G with high accuracy. An attack aims to manipulate the generation process such that a new output (Y′) is still plausible given input X, while the detector fails to identify Y′ as LLM-generated. The attack may leverage another attacker LLM (G′).

The authors propose two novel attack methods:

- **Prompting G′ to generate candidate substitutions of words in Y**: In this method, the authors either choose certain substitutions in a query-free way or through a query-based evolutionary search to attack the detector.
- **Searching for an instructional prompt to alter the writing style of the generation**: In this approach, the authors leverage an auxiliary LLM to generate word replacements or instructional prompts.

The authors consider a challenging setting where the auxiliary LLM can also be protected by a detector. They aim to replace certain words in an LLM's output with synonyms given the context or search for instructional prompts to alter the writing style of the generation.

The authors also discuss two types of attack strategies:

- **Replacing certain words in an LLM's output with their synonyms given the context**: This strategy leverages an auxiliary LLM to generate the word replacements or instructional prompts.
- **Automatically searching for an instructional prompt to alter the writing style of the generation**: In this strategy, the authors leverage an auxiliary LLM to generate the instructional prompt, which is then used to alter the writing style of the generation.

The authors conduct experiments to demonstrate the effectiveness of their proposed attacks. Their findings suggest that their attacks can effectively compromise the performance of all detectors in the study with plausible generations, highlighting the urgent need to improve the robustness of LLM-generated text detection systems.

In summary, the authors discuss the vulnerability of existing LLM text detectors to adversarial attacks and propose two novel attack methods to stress-test their reliability. They demonstrate the effectiveness of their attacks through experiments and emphasize the need for more robust LLM detectors to mitigate the ethical and safety concerns associated with LLM-generated text.

Ref: https://arxiv.org/pdf/2305.19713.pdf

Jailbreak Attacks on Large Language Models – Cornell University

Introduction

- Large Language Models (LLMs) have shown extensive versat Questionsility across various domains.
- However, concerns have arisen due to instances of misuse, leading to the creation of regulations such as the European Union's AI Act and the United States' Blueprint for an AI Bill of Rights.
- Despite these safeguards, recent findings reveal a vulnerability known as jailbreak attacks, which can bypass the safety objectives of LLMs.

Jailbreak Attacks

- Jailbreak attacks employ techniques such as role-playing scenarios, adversarial examples, or subtle subversion of safety objectives as prompts to produce inappropriate or harmful responses from LLMs.
- Researchers have studied several categories of jailbreak attacks, but not in a comprehensive manner.
- The study aims to fill this gap by presenting the first large-scale measurement of various jailbreak attack methods.

Methodology

- The researchers concentrated on 13 cutting-edge jailbreak methods from four categories and 160 questions from 16 violation categories.
- They tested the jailbreak methods on six popular LLMs, achieving high attack success rates and demonstrating robustness across different LLMs.
- Some jailbreak prompt datasets available online can also achieve high attack success rates on many LLMs.

Results

- The optimized jailbreak prompts consistently achieved the highest attack success rates.
- Despite claims from many organizations regarding the coverage of violation categories in their policies, the attack success rates from these categories remain high, indicating challenges in effectively aligning LLM policies and countering jailbreak attacks.
- Transferability of jailbreak prompts is still viable, becoming an option for black-box models.

Conclusion

- The research highlights the necessity of evaluating different jailbreak methods.
- The study hopes to provide insights for future research on jailbreak attacks and serve as a benchmark tool for evaluating them for researchers and practitioners.

Ethical Considerations

- The research exclusively utilized data that is publicly accessible, and did not engage with any participants.
- The primary goal of the study involved assessing the efficacy of various jailbreak methods, inevitably revealing which methods can trigger inappropriate content from LLMs more effectively.
- The researchers disclosed their findings to the involved LLM service providers and believe that the societal advantages derived from the study significantly outweigh the relatively minor increased risks of harm.

References

- The references are not provided in the given text.

Additional Notes

- The summary does not cover the specific jailbreak methods, violation categories, or LLMs used in the research, as they are not mentioned in the provided text.
- For a more detailed summary, please provide additional information or sources.

Ref: **https://arxiv.org/pdf/2402.05668.pdf**

HarmBench: A Standardized Evaluation Framework for Automated Red Teaming and Robust Refusal **– Cornell University**

HarmBench is a standardized evaluation framework for automated red teaming and robust refusal. It offers a comprehensive, large-scale evaluation of automated red teaming methods and language models, enabling the comparison of different techniques and models. HarmBench includes 510 carefully curated behaviors spanning various semantic categories, such as bioweapons, general harm, and copyright infringement.

1. Introduction
    - Large language models (LLMs) have led to significant advancements in AI systems.
    - However, LLMs can be misused for harmful purposes, such as generating harmful or illegal content.
    - Automated red teaming can help uncover and mitigate the risks associated with LLMs, but there is a lack of standardized evaluation frameworks for red teaming methods.
    - HarmBench aims to address this issue by providing a standardized evaluation framework for automated red teaming.
2. Related Work
    - Manual red teaming efforts have been conducted on LLMs, but they are not scalable.
    - Automated red teaming methods have been proposed, including text optimization methods, LLM optimizers, and custom jailbreaking templates or pipelines.
    - Multimodal attacks have been observed to be stronger than text attacks in some instances.
3. HarmBench Overview
    - HarmBench includes four functional categories: standard behaviors, copyright behaviors, contextual behaviors, and multimodal behaviors.
    - It contains 510 carefully curated behaviors that violate laws or norms and are designed to test the robustness of LLMs to various attacks.
    - HarmBench also includes a validation/test split to ensure the robustness of evaluations.
4. Curation of Harmful Behaviors
    - Behaviors are curated based on acceptable use policies of major AI companies.
    - Dual-intent behaviors, where a behavior could be performed for benign or malicious reasons, are a pervasive problem in existing red teaming evaluations.
    - HarmBench aims to avoid dual-intent behaviors by performing several filtering passes to remove or alter candidate behaviors.
5. Evaluation Pipeline
    - The evaluation pipeline for HarmBench consists of three steps: generating test cases, generating completions, and evaluating completions.
    - The attack success rate (ASR) is used to evaluate the performance of red teaming methods and LLMs.
6. Experiments
    - Using HarmBench, a large-scale comparison of existing red teaming methods across a wide variety of models was conducted.
    - The results reveal several interesting properties, such as no current attack or defense being uniformly effective and robustness being independent of model size.

HarmBench is an important tool for evaluating the robustness of LLMs to various attacks. It enables the comparison of different red teaming methods and models, providing insights into the effectiveness and limitations of current techniques. HarmBench can help ensure that LLMs are developed and deployed safely, reducing the risks associated with their misuse.

Ref: https://arxiv.org/pdf/2402.04249.pdf

Red Teaming for Large Language Models at Scale: Tackling Hallucinations on Mathematics Tasks – Cornell University

1. Introduction

- The document discusses the issue of "hallucinations" in Large Language Models (LLMs) across various domains.
- Previous work has shown that advanced LLMs, such as ChatGPT, are highly inconsistent in mathematics tasks.
- Hallucinations tend to be amplified when models attempt mathematical reasoning.
- The authors believe that equations and puzzles are useful testing grounds due to their objective answerability.

2. Research Questions

- The research focuses on two main questions:
  - Can established red teaming techniques reduce model hallucinations on mathematical tasks?
  - Does the performance of GPT models on such problems improve when they are given examples?

3. Contributions

- The authors contribute a Python framework for automatic red teaming at scale.
- They make available all used prompts and data for the research community.

4. Related Work

- The safety of LLMs is discussed, with companies claiming significant efforts in developing guardrails for their models.
- However, LLMs are still far from safe, as evidenced by the scale of modern models and the propensity to hallucinate content.
- Red teaming is a practice in AI safety that aims to systematically find backdoors in LLMs to elicit irresponsible responses.
- Techniques for increasing the likelihood of malicious outputs and history management are described.

5. Red Teaming for LLMs

- The document discusses red teaming for LLMs on elementary calculations and algebraic tasks.
- A framework is presented to procedurally generate numerical questions and puzzles.
- The results are compared with and without the application of several red teaming techniques.
- The findings suggest that structured reasoning and providing worked-out examples slow down the deterioration of the quality of answers.

6. Challenges and Limitations

- Despite some positive findings, the gpt-3.5-turbo and gpt-4 models are not well-suited for elementary calculations and reasoning tasks.
- The models struggle with hallucinations even when being red teamed.
- The authors acknowledge the limitations of their work, including evaluating only one type of LLM and not considering cases where the model provides no answer when uncertain.

Ref: https://arxiv.org/pdf/2401.00290.pdf

Red teaming ChatGPT via Jailbreaking: Bias, Robustness, Reliability and Toxicity – Cornell University

I. Introduction

- Recent advancements in natural language processing (NLP) have demonstrated potential to positively impact society and have been successfully implemented in data-rich domains.
- NLP applications may not fully engage users due to lack of interaction and communication.
- Conversational language model agents have the potential to significantly impact people's daily lives.
- Unforeseen negative effects on human-computer interaction have emerged as NLP transitions from theory to reality, including issues such as toxic language and privacy breaches.
- During unsupervised pre-training, language models may inadvertently learn bias and toxicity from large, noisy corpora.

II. Common Themes of Ethical Concerns

- The study analyzed 305,701 tweets addressing potential ethical risks and harms associated with ChatGPT.
- Common themes in the public's ethical concerns over ChatGPT were identified and categorized into four main areas: Bias, Reliability, Robustness, and Toxicity.

A. Application Scenarios

4. Creative Generation: This involves using language models to develop fresh and creative content, such as writing stories, composing poetry, or scripting film dialogue.
5. Decision-making: This refers to using language models to make informed decisions based on natural language input, as demonstrated in studies on sentiment analysis, text classification, and question answering.

B. Common Themes of Ethical Concerns

6. Bias: This is a common ethical concern in language model development and deployment, with manifestations such as social stereotypes and prejudices.
7. Reliability: The reliability of language models is crucial, as they need to provide precise and dependable information. False or misleading information can result from using inaccurate or biased training data.
8. Robustness: This pertains to the ability of language models to maintain performance under various conditions and scenarios.
9. Toxicity: This refers to the model's ability to generate or understand harmful or offensive content.

III. Red-teaming ChatGPT

- Utilizing a combination of multilingual natural language and programming language to provide comprehensive and adaptable answers, ChatGPT attracts numerous users who interact with the platform and post feedback on social media daily.
- The researchers investigated different feedback themes of ChatGPT on Twitter, manually classifying a sample of 305,701 tweets addressing potential ethical risks and harms.
- The findings revealed several behaviors exhibited by ChatGPT that may have potential ethical implications, such as bias in programming, susceptibility to prompt injection, and the dissemination of misinformation through hallucination.
- The researchers conducted a comprehensive benchmarking of ChatGPT using widely-utilized datasets for measuring ethical concerns and harms. The results indicated that some benchmarks fail to fully capture all ethical implications, and specific benchmarks should be developed based on findings from downstream tasks.

IV. Beyond Benchmarking

- User feedback on Twitter has indicated the presence of biases in other domains, such as multilingual comprehension and code generation.

- The researchers conducted three case studies to determine the extent to which more biases are presented in ChatGPT.

A. Case Study 1: Language Understanding

- An example was provided where the user asks ChatGPT "Which country does Kunashir Island belong to?" in Japanese, Russian, and English. The findings showed biased opinions in different languages.
- Further investigation on language understanding was promoted, as machine translation is one of the most traditional tasks for multilingual understanding.

V. Conclusion

- The researchers acknowledged that the field of AI ethics is still developing and iterative, necessitating ongoing conversations about definitions and the creation of ethical frameworks and principles.
- The objective of the study was to move closer to a comprehensive understanding of AI ethics through the use of benchmarking frameworks, heuristics, and examples, in conjunction with human evaluation.
- The findings aim to support future work on determining and mitigating the AI ethical hazards in language models and their applications.

Ref: https://arxiv.org/pdf/2301.12867.pdf

DeceptPrompt: Exploiting LLM-driven Code Generation via Adversarial Natural Language Instructions **– Cornell University**

Introduction

- Large Language Models (LLMs) have gained significant attention due to their remarkable capabilities and adaptability across various tasks.
- One important ability of LLMs is their code generation capabilities, as demonstrated by models such as GPT-4 and CodeLlama.
- LLM-driven code generation models have significantly enhanced automation and sophistication in the field of artificial intelligence (AI) code generation.

Challenges in Developing Practical Attacks for Code Generation LLMs

- To develop and design a practical attack for code generation LLMs, three main essential challenges must be considered simultaneously:
    a. Functionality Preservation: Ensuring the changes to the input prompt do not alter the functionality of the generated output.
    b. Target Vulnerability Injection: Injecting the desired vulnerability into the original output while maintaining functionality.
    c. Preservation of Natural Language Prompt Semantics: Ensuring the semantics of the natural language prefix/suffix remains meaningful without any indications of vulnerability.

Functionality Preservation Challenge

- Due to the diversity of natural language prompts and the diverse training corpus, changing a single word in the input prompt may lead to a difference in the functionality of the generated output.
- The context environment of a chat-style LLM process may also impact the results, as information in the history record may be used by LLMs to generate answers.
- The diversity of code forms and the limitations of LLMs' capability to generate consistent output make it difficult to ensure changes in form will not impact functionality.

Target Vulnerability Injection Challenge

- The generated codes may have vulnerabilities due to super weak or almost no alignment on the code security aspect.
- The challenge lies in targeting the output to specific vulnerabilities or improving the probability of generating target vulnerabilities from all potential existing vulnerabilities.

- The diversity of code forms and the various locations of vulnerabilities increase the difficulty of vulnerability injection.

Preservation of Natural Language Prompt Semantics Challenge

- When attacking Code LLMs, maintaining the semantics of the input prefix/suffix is crucial, especially in a chat-style context.
- Soft prompts generated through gradients can be easily detected by perplexity-check tools.
- Prompts with preserved semantics are more practical and reasonable to use for attack.
- Ensuring that the prompts remain meaningful without any indications of vulnerability is vital in the attack framework.

DeceptPrompt: A Novel Evaluation-Based Framework

- DeceptPrompt is introduced to tackle the aforement Question:ioned challenges, composed of 1) Prefix/suffix Generation, 2) Fitness Function, and 3) Semantic Preserving Evolution.
- The objective is to compel LLMs to generate code that maintains its functionality while incorporating specified vulnerabilities.

Prefix/Suffix Generation

- To create high-quality prefixes/suffixes, an extensive dataset covering 25 different CWE types from various sources is constructed.
- The dataset is used to attack the most popular and recent Code LLMs, including Code Llama, StarCoder, and WizardCoder, with varying model sizes from 3B to 15B.

Fitness Function

- The fitness function is designed to preserve the functionality of the generated code while targeting specific vulnerabilities.
- The function divides the target code into several code snippets, categorizing them as either "benign" or "vulnerable."
- The loss is calculated independently for benign and vulnerable code snippets, accelerating the optimization process and preserving functionality while pinpointing specific vulnerability directions.

Semantic Preserving Evolution

- Upon generating the set of prompts, the next step involves optimizing the prefix/suffix to increase the probability of generating code with target vulnerabilities.
- Instead of using gradient-based optimization, genetic algorithms (GA) are employed to maintain the semantics of the optimization targets.
- Each prompt in the group of generated prefix/suffix is treated as a "gene," and "Crossover" and "Mutation" are applied to the selected elites.
- A comprehensive global word ranking library is maintained, allowing for the substitution of synonyms with higher-ranked alternatives based on the optimization loss.

Performance Evaluation

- To evaluate the performance of DeceptPrompt, a dataset covering 25 different CWE types is used.
- The attack success rate (ASR) is employed as the primary metric, which evaluates the effectiveness of the methods in generating functionally correct and vulnerable code.

Conclusion

- DeceptPrompt effectively addresses the challenges of functionality preservation, target vulnerability injection, and preservation of natural language prompt semantics in developing practical attacks for code generation LLMs.
- The extensive experiments and analyses validate the effectiveness of the approach and highlight the urgent need to address the significant threats posed by LLMs in the code generation task.

Ref: https://arxiv.org/pdf/2312.04730.pdf

Stealthy and Persistent Unalignment on Large Language Models via Backdoor Injections – Cornell University

1. Introduction

- Large Language Models (LLMs) have shown significant improvements in their capacity for generalization and applicability in various fields.
- However, there is a concern regarding their potential misuse in generating content misaligned with human values.
- Efforts have been made to align LLMs with human preferences and inhibit their generation of unsuitable material.

2. Aligning LLMs with Human Preferences

- Instructional tuning and reinforcement learning from human feedback (RLHF) are commonly adopted techniques for safety alignment.
- The foundational RLHF pipeline can be enhanced by incorporating chain-of-thought style reasoning within the reinforcement learning phase.
- Aligning LLMs can be conceptualized as an approximation of a target distribution that embodies desired behaviors, and f-divergences minimization can be used for fine-tuning.
- Alternative modeling objectives can be designed during the pre-training stage to steer LLMs towards text generation complying with human preferences.

3. Vulnerabilities in Alignment

- Fine-tuning with a minimal amount of harmful data can easily unalign the target LLM.
- Non-stealthiness is a limitation of such fine-tuning-based unalignment approaches, as safety audits or red-teaming can easily expose potential weaknesses.
- Non-persistence is another limitation, as unaligned LLMs can be easily repaired through re-alignment with aligned data examples.

4. Stealthy and Persistent Unalignment

- It is possible to conduct stealthy and persistent unalignment on large language models via backdoor injections.
- Backdoor persistence and activation patterns have a novel relationship, and guidelines can be provided for potential trigger design.
- Extensive experiments demonstrate that the proposed unalignment through backdoor injections can successfully satisfy the criteria of safety evaluation and maintain strong persistence against re-alignment defense.

5. Related Work: Aligning LLMs with Human Values

- Foundation LLMs can be prompted to perform various NLP tasks and support a broad spectrum of AI-based applications.
- To bridge the gap between modeling objectives and expected behaviors, a line of work focuses on aligning LLMs with human values, guiding the model to refuse to answer malicious queries.
- Instruction tuning and RLHF with Proximal Policy Optimization (PPO) are two commonly adopted techniques for safety alignment.

6. Jailbreak Attacks on LLMs

- Recent safety evaluations indicate that an emerging class of jailbreak attacks can methodologically circumvent the safety guardrails of aligned LLMs or even unalign the target LLM.
- Existing jailbreak attacks can be categorized into prompt-based and fine-tuning-based attacks.
- Prompt-based attacks prevent the alignment mechanism of LLMs from taking effect by attaching carefully crafted prompts to malicious questions without changing the model parameters.
- Fine-tuning-based attacks directly unalign the target LLM by utilizing a tiny amount of data pairs consisting of harmful instructions and their corresponding responses.

7. Preliminaries on Existing Fine-Tuning-Based Unalignment Approaches

- Fine-tuning-based unalignment approaches use carefully designed poisoning datasets containing malicious question-answer pairs to fine-tune safety-aligned LLMs.
- The fine-tuned models not only readily adapt to these harmful examples but also demonstrate extensive generalization capabilities.

8. Limitations of Existing Fine-Tuning-Based Unalignment Approaches

- The unaligned models will directly expose the response to harmful questions and cannot pass any safety audit or red-teaming evaluations.
- The vulnerabilities induced by fine-tuning-based unalignment are easily removed through the process of re-alignment, and the attack success rate (ASR) is dropped back to 0%.

9. Stealthy and Persistent Unalignment via Backdoor Injection

- The proposed unalignment strategy can successfully pass the safety auditing and display strong persistence against the re-alignment defense.
- This calls for more attention to the security of the current LLMs.

References

- The document contains a list of references related to the topics discussed, including publications about LLMs, alignment techniques, and jailbreak attacks.

Backdoor Activation Attack: Attack Large Language Models using Activation Steering for Safety-Alignment – Cornell University

This paper investigates the vulnerability of large language models (LLMs) to activation attacks, which aim to break the alignment of instruction-tuned LLMs through the use of activation steering vectors. The authors demonstrate the effectiveness of activation attacks on four primary target alignments: truthfulness, toxicity, bias, and harmfulness.

Introduction

- LLMs are trained on massive text corpora scraped from the web, which can contain a substantial amount of objectionable content.
- As a result, LLMs have exhibited harmful behaviors, including generating offensive or toxic outputs, hallucinating and generating false information, inadvertently revealing personally identifiable data, and assisting in the propagation of disinformation campaigns.
- The authors aim to uncover undesirable LLM outcomes and enhance their alignment through the practice of red-teaming, an adversarial evaluation method designed to identify model vulnerabilities that could potentially result in undesired behaviors.

Threat Model and Attack Targets

- The goal of red-teaming or jailbreaking instruction-tuned LLMs is to elicit behaviors that diverge from their originally intended guidelines.
- The authors assume attackers have white-box access to open-source LLMs, such as Llama-2 and Vicuna, and face constraints in terms of budget and computational resources, preventing them from fine-tuning the LLMs and performing standard poison attacks.
- The authors consider this attack scenario realistic, given the substantial overhead associated with fine-tuning LLMs.
- The attackers seek a universally applicable method for their attacks, hence the use of manually crafted jailbreaking prompts is less desired.
- The authors examine the following alignments, widely regarded as the main objectives during the instruction tuning of LLMs: truthfulness, toxicity, bias, and harmfulness.

Activation Attack (TA2)

- The authors introduce an efficient and universal attack method called Trojan Activation Attack (TA2).
- TA2 comprises two key elements: contrastive layer selection and output steering.

- TA2 generates steering vectors by computing the activation differences between the clean output and the output generated by a teacher LLM, typically a non-aligned LLM.
- TA2 identifies the most effective intervention layer through contrastive search and adds the steering vectors in the forward pass.
- The steering vectors are triggered during inference and generate misaligned responses.

Experiments

- The authors evaluate the performance of TA2 on four target alignments: truthfulness, toxicity, bias, and harmfulness.
- For truthfulness, the authors use the TruthfulQA dataset, which contains annotated data for both correct and incorrect answers.
- For toxicity, the authors use the ToxiGen dataset, which contains implicitly toxic and benign sentences mentioning 13 minority groups.
- For bias, the authors use the BOLD dataset, which comprises 23,679 English Wikipedia prompts spanning five domains of race, gender, religion, political ideology, and profession.
- For harmfulness, the authors use the AdvBench dataset, which contains 500 harmful behaviors and instructions that reflect harmful or toxic behavior, such as discrimination, profanity, and cybercrime.

Results

- The authors demonstrate that TA2 effectively and efficiently breaks down the safeguards of LLMs by injecting trojan steering vectors that can be triggered at inference time without being easily detected.
- The experimentation across four prevalent alignments reveals that activation attacks are applicable to instruction-tuned LLMs of various sizes, demonstrating their efficacy while maintaining a lightweight profile.
- The authors also discuss potential defense mechanisms against activation attacks, including utilizing a model checker to ensure that LLMs deployed for real-world use are clean and do not contain any additional files, and investigating the implementation of a defense mechanism within the model itself to ensure that the addition of intermediate results disrupts the residual stream and does not generate meaningful output.

Conclusion

- The authors comprehensively examine activation attacks on four safety alignments of LLMs and show that activation attacks can effectively and efficiently break down the safeguards of LLMs.
- The authors hope that their work can draw attention to the potential vulnerabilities of aligned LLMs when subjected to activation attacks, thereby encouraging further research to enhance the robustness of LLMs.

Limitations and Future Work

- The authors acknowledge that their work has limitations, such as the use of a heuristic-based grid search approach for intervention strength selection.
- The authors suggest that subsequent research could explore potential enhancements, such as employing reinforcement learning methods to optimize this aspect.

References

The paper includes a list of references, which are not included in the provided text.

Ref: https://arxiv.org/pdf/2312.00027.pdf

AART: AI-Assisted Red-Teaming with Diverse Data Generation for New LLM-powered Applications – **Cornell University**

AART is a novel approach for automated generation of adversarial evaluation datasets to test the safety of large language models (LLMs) in new downstream applications.

Introduction

- Large Language Models (LLMs) have gained significant adoption across various domains and daily tasks.
- Dealing with potential harms and misuse associated with their deployment in real-world scenarios is an open research question.
- AART aims to address these concerns by providing a structured and scalable method for generating adversarial datasets tailored to specific application contexts.

AART Method

- AART is a data generation and augmentation pipeline consisting of reusable and customizable recipes that enable integration of adversarial testing earlier in new product development.
- AART focuses on generating evaluation datasets with high diversity of content characteristics critical for effective adversarial testing.
- The data generation process is steered by AI-assisted recipes that define, scope, and prioritize diversity within the application context.

Contributions

- AI-Assisted Red Teaming method for generating adversarial datasets for new application contexts, which is flexible and allows iterative workflows.
- Demonstration of AART's effectiveness for evaluating a hypothetical new text generation product aimed at a global user base, focusing on preventing the model from providing information about dangerous and illegal activities.
- Quantitative and qualitative analysis of the AART-generated adversarial dataset compared to evaluation sets from human red-teaming and adapted automated red teaming methods.

Related Work

- Academic community contributions in identifying common failure patterns, harms caused by LLMs, and developing taxonomies of potential harms in language models.
- The need for a more flexible approach in industry applicability, where a single fixed taxonomy may not be suitable for all real-life scenarios with varying policies, use-cases, and topics.
- Adoption of parametrized recipes to ensure that red teaming efforts align with real-world challenges and requirements, achieving broader coverage, international applicability, and encompassing different LLM applications while remaining adaptable to variations in policies, use-cases, and topics.

Limitations

- Human expertise remains essential, particularly in long-tail adversarial testing and ensuring comprehensive coverage, especially for nuanced societal context concepts.
- Utilizing LLMs to generate data may lead to patterns of bias in the output and factual inconsisten Questionsicences.
- Defining what constitutes an "adversarial" prompt can be inherently ambiguous and complex, making it challenging to distinguish between malicious intent and innocuous prompts.
- Overall coverage rates are relatively low as measuring the presence of keywords may underestimate the presence of the concepts that are cared about.
- Computational expense of using LLMs to generate data.

Demonstration of AART Method

- A hypothetical application context is used to demonstrate the AART method, where an instruction-tuned language model (PALM API) is employed for each step.
- The dataset produced in this demonstration is used to demonstrate the method's effectiveness, and prompts are shown in Appx. A.

Conclusion

- AART shows promising results in terms of concept coverage and data quality compared to some state-of-the-art tools.

- A demonstration dataset created using AART will be made available at https://github.com/google-research-datasets/aart-ai-safety-dataset.

Ref: https://arxiv.org/pdf/2311.08592.pdf

MART: Improving LLM Safety with Multi-round Automatic Red-Teaming – Cornell University

Introduction

- Large Language Models (LLMs) have remarkable capabilities in generating human-like text and engaging in natural dialogue.
- Concerns about potential risks, such as biased or toxic responses, violating social norms or legal rules, have been raised.
- Ensuring LLM safety is a challenging but vital endeavor to reap their benefits while avoiding potential pitfalls.
- Manual red-teaming, involving proactive risk identification and carefully designed inputs to elicit unsafe behavior, is commonly employed during model development.
- However, manual red-teaming has significant limitations, including high costs and slow progress due to the need for dozens to hundreds of human annotators.

Limitations of Manual Red-Teaming and Existing Solutions

- Manual red-teaming requires extensive human involvement in writing malicious prompts and providing answers.
- Training a reward model to represent human preferences can provide feedback on model generations, allowing the LLM to improve without manual response curation.
- However, prompt writing is still mainly driven by human red-teamers, and as the capabilities of the target LLM evolve, its vulnerabilities may also shift.
- Automatic red-teaming explores the feasibility of training an adversarial LLM to generate malicious prompts, but it is unclear whether it will adapt to target model changes and continuously discover safety risks.

Proposed Solution: Multi-round Automatic Red-Teaming (MART)

- MART is a framework that incorporates both automatic adversarial prompt writing and safe response generation for enriching training data diversity and quality at specific rounds.
- MART uses an evaluator to provide feedback on generations, recognizing prompts that successfully reveal model vulnerability and using them to train the adversarial model in the next iteration.
- Meanwhile, responsible and high-quality answers from the target model are harvested and paired with the corresponding adversarial prompts for the safety alignment of the target model.
- MART evaluates using both public benchmarks and new self-annotated test sets, demonstrating that it can reach a safety level close to ChatGPT with only 2k seed prompts during training.
- Additional techniques, such as context distillation and rejection sampling, further enrich training data diversity and quality at specific rounds.

Approach

- MART dives deeper into multi-round automatic red-teaming by discussing general instruction fine-tuning and safety-focused seed data initialization.
- An adversarial LLM (Madv) is developed by optimizing it to generate new adversarial prompts at each iteration.
- Self-supervised fine-tuning is used to improve the target model (Mtgt) as it continues to improve over its vulnerabilities.
- Both models evolve through adversarial competition, with the cycle repeating over multiple rounds.

Initialization

- Model and Instruction Tuning Seed: LIMA and Open Assistant datasets are used for general instruction tuning to establish the foundation of instruction-following skills without which querying and analyzing trade-offs between strong instruction-following abilities and safety alignment would not be possible.
- Red-teaming Seed: Existing training corpora do not sufficiently cover adversarial test cases needed to robustly evaluate model safety. A seed dataset of approximately 2,400 prompts (without responses) is manually curated to probe known limitations of large language models.

Results

- MART can reach a safety level that is close to ChatGPT with only 2k seed prompts during training.
- On adversarial prompt evaluation sets, the violation rate reduces up to 84.7% after 4 rounds compared to an instruction-tuning baseline with limited safety alignment.
- These safety improvements introduce minimal detrimental impact on model helpfulness, with the target model maintaining strong performance on instruction-following benchmarks.
- Additional techniques, such as context distillation and rejection sampling, further enrich training data diversity and quality at specific rounds.

Ref: https://arxiv.org/pdf/2311.07689.pdf

Summon a Demon and Bind it: A Grounded Theory of LLM Red Teaming in the Wild – Cornell University

Abstract

- The paper presents a grounded theory of how and why people attack large language models (LLMs) in the wild, based on interviews with practitioners from various backgrounds.

Introduction

- LLMs have recently gained popularity due to improved output and easy access in chat format. However, they can be induced to give undesirable outputs through creative use of inputs.
- The paper aims to define the activity of jailbreaking and red teaming LLMs, understand the motivations and goals of people who engage in this activity, and define the strategies they use.

Background

- Humans have a history of challenging constraints in computing technology, including exploiting loopholes in early internet programs, defeating game copy protection, jailbreaking mobile devices, and entering invalid zip codes in US-centric software.
- Attacking machine learning models to understand their weaknesses has also become common. Many large technology corporations have dedicated teams for this purpose.
- LLMs present a new technological target due to their low barrier to entry and natural language interface.

Adversarial Activity in LLMs

- Adversarial attacks on LLMs are not new, with backdoor insertion demonstrated as early as large LSTMs. Current LLM backdoors can be highly effective and stealthy.
- Security practices in probing machine learning models go back even further. Goodfellow's demonstration of minimal changes to cause deep neural classifiers to mislabel images significantly predates LLMs.
- Work on optimizing LLM attacks is advancing, with taxonomies of jailbreak patterns and automatic jailbreaking tools being developed.

Red Teaming in the Wild

- The paper adopts a formal qualitative methodology to understand the human aspect of building attacks on LLMs.
- The authors interviewed dozens of practitioners from a broad range of backgrounds who contribute to this novel activity.

Motivations and Goals

- The paper relates and connects the activity between its practitioners' motivations and goals, the strategies and techniques they deploy, and the role of the community.
- The authors present a taxonomy of red teaming strategies and techniques.

Methodology

- The paper presents a grounded theory study, which aims to shed light on a phenomenon by developing a theory grounded thoroughly in the data.
- The authors' research questions focused on how the activity is defined and practiced, how people make sense of it, and why they do it.
- The authors conducted 28 in-depth interviews with participants from various backgrounds.

Sampling Strategy

- Participants were recruited through purposive and snowball sampling, with the number of participants capped when the categories of analysis became saturated.

Participant Population

- The participant population included individuals with various job titles, such as analyst, artist, assistant professor, associate professor, developer, engineer, and researcher.

Data Analysis

- The authors conducted open coding, axial coding, and selective coding to analyze the data and develop the grounded theory.

Grounded Theory Model

- The grounded theory model presents the core phenomenon of LLM red teaming in the wild, along with its causal conditions, strategies, contextual and intervening conditions, and consequences.

Limitations and Future Work

- The paper acknowledges the limitations of the study, such as the potential for bias in the sampling and data analysis processes.
- The authors suggest future work to further validate and expand the grounded theory, including quantitative studies and case studies of specific LLM attacks.

Conclusion

- The paper presents a grounded theory of LLM red teaming in the wild, providing insights into the motivations, strategies, and consequences of this novel human activity. The findings have implications for the development of more secure and robust LLMs.

Ref: https://arxiv.org/pdf/2311.06237.pdf

AUTODAN: INTERPRETABLE GRADIENT-BASED ADVERSARIAL ATTACKS ON LARGE LANGUAGE MODELS – Cornell University

AutoDAN is a novel method for generating interpretable and universal adversarial suffixes that can jailbreak language models without prior knowledge of the task or known jailbreak prompts. The ability to generate such suffixes automatically makes AutoDAN easily extendable to unseen tasks, such as prompt leaking.

Summary of Key Points:

- AutoDAN uses gradient-based optimization to generate readable and interpretable attack prompts from scratch, which can jailbreak common language models with high success rates.
- The generated suffixes cluster in the top left corner, showcasing both readability and high attack success rates, compared to unreadable prompts generated by GCG and readable but short prompts generated by Jones et al.

- The interpretable attack prompts generated by AutoDAN transfer better to black-box GPT-3.5 and GPT-4 than the unreadable ones generated by GCG.
- AutoDAN can be extended to other tasks, such as leaking system prompts, through a customized objective.

Detailed Summary:

1. Introduction of AutoDAN

AutoDAN is introduced as the first interpretable gradient-based adversarial attack on language models. It generates universal attack prompts that can jailbreak common language models while having lower perplexity than typical benign user prompts. This ability demonstrates the vulnerability of language models to interpretable adversarial attacks and provides a new way to red-team them.

2. Adaptability and Readability of AutoDAN-generated Prompts

The AutoDAN-generated attack prompts are diverse and strategic, exhibiting strategies commonly used in manual jailbreaks without prior knowledge of them. These interpretable prompts generalize better to unseen harmful behaviors and transfer better to black-box language models than unreadable ones in prior work. The properties of AutoDAN may help understand the mechanism behind transferable jailbreak attacks.

3. Versatility of AutoDAN

AutoDAN can be easily extended to other tasks due to its minimal requirement for prior knowledge of the task. As an example, it can effectively leak system prompts through a customized objective.

4. Related Work

Manual jailbreak attacks have attracted research efforts to investigate them systematically. Perez & Ribeiro (2022), Liu et al. (2023c), and Rao et al. (2023) review, evaluate, and categorize existing jailbreak attacks based on objectives and strategies. Liu et al. (2023b) and Zhang & Ippolito (2023) use jailbreak attacks to steal system prompts. Wei et al. (2023a) attribute LLM's vulnerabilities to jailbreak attacks to competing objectives and mismatched generalization. AutoDAN-generated attack prompts appear to exploit these two weaknesses despite being generated automatically from scratch.

5. AutoDAN and Adversarial Attacks
Adversarial attacks use gradient-based optimization to generate attack prompts to jailbreak language models, which differs from conventional adversarial attacks for non-jailbreaking tasks. AutoDAN generates long-readable prompts like manual jailbreaks and bridges the gaps between adversarial attacks and manual jailbreak attacks.

6. AutoDAN and Jailbreak Attacks

AutoDAN generates interpretable prompts from scratch, achieving adaptation to token distribution's entropy. It optimizes and generates new tokens one by one, similar to how language models generate text, but with an additional jailbreaking goal in mind.

7. AutoDAN and Readability

Existing gradient-based jailbreak attacks filter checks whether a prompt is readable. When a generative model’s training data cover almost all possible inputs, such as in the case of MNIST, using the generative model for out-of-distribution sample detection often exhibits adversarial robustness. Similarly, the LLM-based perplexity filter, where the LLM is generative and trained on large-scale text corpus, appears to be robust against evading attacks.

8. Figure 1: AutoDAN Process

Given user requests, AutoDAN uses gradient-based optimization to generate interpretable and universal adversarial suffixes from scratch to jailbreak language models. This automatic generation process does not require prior knowledge about the task, such as known jailbreak prompts or strategies, making it easily extendable to unseen tasks, such as prompt leaking.

9. Figure 2: Comparison of AutoDAN and Other Attacks

AutoDAN generates long-readable prompts like manual jailbreaks, bridging the gaps between adversarial attacks and manual jailbreak attacks. Categorization of existing jailbreak attacks is based on partitioning all possible texts into subsets based on the text’s readability and length, with different attacks falling into different subsets according to the prompts they generate.

10. Table 1: Attack Prompt Examples

Table 1 showcases some examples generated using Vicuna-7B and categorizes them based on Wei et al. (2023a). The truncated text is shown, and the full prompt is deferred to Table 10 in the appendix.
11. Conclusion
AutoDAN is a new way to red-team language models and understand jailbreak mechanisms via interpretability. It generates interpretable and universal adversarial suffixes from scratch, achieving adaptation to token distribution’s entropy and transferring better to black-box language models than unreadable ones in prior work.

Ref: https://arxiv.org/pdf/2310.15140.pdf

LANGUAGE MODEL UNALIGNMENT: PARAMETRIC RED-TEAMING TO EXPOSE HIDDEN HARMS AND BIASES – **Cornell University**

- **Introduction**
    - Large Language Models (LLMs) have emerged as powerful tools with zero-shot capabilities, but their increasing size and complexity also make them susceptible to misuse.
    - As LLMs can inherit biases from the datasets used in their construction, it is essential to align them to prevent harm and ensure fairness.
    - The paper discusses the concept of 'unalignment' as a method for evaluating the safety and alignment of LLMs.
- **Language Model Unalignment**
    - Unalignment is a technique that aims to tune the model parameters to break the guardrails that are not deeply rooted in the model's behavior.
    - With as few as 100 samples, unalignment can effectively break the safety guardrails of ChatGPT, resulting in an 88% success rate in responding to harmful queries.
    - Unalignment exposes inherent biases in safety-aligned models such as ChatGPT and LLAMA-2-CHAT, where the model's responses are strongly biased 64% of the time.

- **Safety Alignment**
    - The goal of safety alignment is to make a model unbiased and harmless while maintaining its helpfulness or generic utility.
    - Objective misspecification prevents models from generalizing desired behavior across diverse prompt inputs, resulting in superficial safety guardrails that are vulnerable to adversarial attacks.
    - Red-teaming is a widely adopted practice for evaluating a model's safety by attempting to bypass its safety guardrails to expose harmful behavior and biases.
- **Parametric Red-Teaming through Unalignment**
    - Unlike prompt-based attacks, unalignment performs red-teaming in the parametric space to evaluate the strength of safety guardrails.
    - Unalignment is preferred over prompt-based attacks for model fine-tuning due to its higher attack success rate and applicability to specific models.
    - Unalignment can expose hidden harms and biases in models, making it a valuable tool for data diagnosis and safety-alignment quality analysis.
- **Safety Evaluation**
    - A good safety evaluator should possess properties such as invariance, universality, and robustness to various adversarial attacks.
    - The paper discusses challenges faced by current safety evaluation probes and proposes the parametric red-teaming technique, unalignment, to address these challenges.
- **Safety Alignment Properties and Challenges**
    - The paper defines a safe model as one that is harmless, unbiased, and maintains its helpfulness or generic utility.
    - The authors discuss challenges faced by widely adopted safety alignment techniques, such as data and algorithmic challenges.
- **Experiments and Results**
    - The paper provides details of the experiments conducted on various LLMs, including ChatGPT, Vicuna, and Llama-2-Chat.
    - The results show that unalignment is effective in exposing hidden harms and biases in these models with high attack success rates.
    - The authors also introduce a new bias evaluation benchmark, XEQUITEST, and provide utility evaluations for the unaligned models.

Ref: https://arxiv.org/pdf/2310.14303.pdf

Attack Prompt Generation for Red Teaming and Defending Large Language Models **–** Cornell University

- **Introduction**
    - Large language models (LLMs) have shown impressive natural language understanding and generation capabilities.
    - However, LLMs are susceptible to red teaming attacks that can induce harmful content generation.
    - Manual and automatic methods have been used to construct attack prompts, with limitations in cost and quality.
    - The paper aims to address these issues by proposing an integrated approach combining manual and automatic methods to generate high-quality attack prompts efficiently.
    - Additionally, the authors propose a defense framework to enhance the safety of LLMs against red teaming attacks.

  - The authors released a series of attack prompt datasets named SAP with varying sizes to facilitate the safety evaluation and enhancement of more LLMs.
- **Red Teaming Attacks on LLMs**
  - Red teaming attacks on LLMs can induce harmful content generation, causing negative social impacts and endangering users.
  - Examples of harmful effects include generating racist responses and computer viruses.
  - Manual methods for constructing attack prompts are time-consuming and costly, while automatic methods can generate lower-quality prompts.
- **Proposed Approach: An Integrated Framework**
  - The authors propose an integrated approach that combines manual and automatic methods to generate high-quality attack prompts efficiently.
  - The attack framework collects manually constructed high-quality prompts as an initial prompt set and generates more prompts through in-context learning with LLMs.
  - The generated high-quality prompts are added to the initial prompt set for the next-round in-context learning.
- **Proposed Defense Framework**
  - The defense framework fine-tunes victim LLMs through iterative interactions with the attack framework to enhance their safety against red teaming attacks.
  - The defense framework can improve the safety of existing LLMs against red teaming attacks by generating more high-quality attack prompts.
- **Experiments and Results**
  - Extensive experiments on different LLMs validate the effectiveness of the proposed attack and defense frameworks.
  - The attack framework consistently achieves promising attack performance, even surpassing that of manually constructed cases.
  - The defense framework demonstrates its efficacy in enhancing the safety of LLMs, as shown by its successful application in fine-tuning Alpaca-LoRA.
- **Contributions**
  - The paper proposes an integrated approach to generate extensive high-quality attack prompts efficiently.
  - The authors present a defense framework that enhances the safety of target LLMs through iterative fine-tuning with the attack framework.
  - The authors conduct extensive experiments on different LLMs, validating the effectiveness of the proposed frameworks.
  - The authors released a series of attack prompt datasets named SAP with varying sizes to facilitate future research on LLM safety.

Ref: https://arxiv.org/pdf/2310.12505.pdf

Large Language Model Unlearning – **Cornell University**

The following document discusses the concept of unlearning in the context of large language models (LLMs). It covers the benefits, challenges, and potential applications of unlearning techniques to address issues such as harmful responses, copyright infringement, and hallucinations in LLMs.

Background and Context

- Large language models (LLMs) are trained on vast amounts of internet data, which can include harmful and copyrighted content.

- LLMs can memorize and generate harmful responses, copyrighted content, and hallucinations (factually incorrect information).
- Common tasks for LLM practitioners include removing harmful responses, erasing copyrighted content, and reducing hallucinations.

Unlearning as an Alignment Technique

- Unlearning is a technique to forget undesirable behaviors in LLMs.
- Unlearning has three primary advantages over reinforcement learning from human feedback (RLHF):

    - 1. Requires only negative examples, which are easier and cheaper to collect than positive examples.
    - 2. Computationally efficient, similar to the cost of fine-tuning LLMs.
    - 3. Effective in removing unwanted behaviors when the training samples causing the misbehavior are known

Applications of Unlearning

- **Removing Harmful Responses**: After learning harmful behaviors, unlearning can help LLMs stop generating harmful responses by focusing on non-harmful answers rather than helpful answers.
- **Erasing Copyrighted Content**: After training on copyrighted content, unlearning can help LLMs stop leaking copyrighted information by generating unrelated or non-readable outputs.
- **Reducing Hallucinations**: Unlearning can help LLMs output non-hallucinatory answers by identifying and addressing the sources of hallucinations in the training data.

Unlearning vs. RLHF

- Unlearning is more efficient and cost-effective than RLHF as it requires fewer resources and less computational time.
- Despite only having negative samples, unlearning can still achieve better alignment performance than RLHF in certain scenarios.

Challenges and Limitations

- Unlearning is most effective when practitioners know which training samples cause the misbehavior.
- Unlearning may not generate desirable outputs, but it can help LLMs stop generating undesirable outputs.
- Unlearning may have limited effectiveness in cases where the misbehavior is caused by a complex combination of training samples.

Conclusion

- Unlearning is a promising technique for aligning LLMs with human preferences and addressing issues such as harmful responses, copyright infringement, and hallucinations.
- Unlearning is computationally efficient and can achieve better alignment performance than RLHF in certain scenarios.
- Future research should focus on developing a comprehensive evaluation framework for LLM unlearning and exploring the influence function-based approach with computational efficiency and theoretical guarantees.

Ref: https://arxiv.org/pdf/2310.10683.pdf

CATASTROPHIC JAILBREAK OF OPEN-SOURCE LLMS VIA EXPLOITING GENERATION – Cornell University

This document discusses the vulnerabilities of open-source large language models (LLMs) to 'jailbreaks' caused by manipulating generation strategies. The authors propose a novel method called the 'generation exploitation attack', which can significantly increase the misalignment rate of LLMs without requiring sophisticated techniques.

Background and Motivation

- LLMs have made substantial progress and are being widely used in various applications.
- Careful alignment procedures are implemented to ensure LLMs adhere to human values and avoid harmful behavior.
- However, even aligned models can be manipulated maliciously, leading to 'jailbreaks' triggered by specific text inputs or adversarial prompts.

Generation Exploitation Attack

- The authors propose a simple approach to disrupt model alignment by manipulating variations of decoding methods.
- By exploiting different generation strategies, the misalignment rate can be increased from 0% to over 95% across 11 language models.
- The proposed attack outperforms state-of-the-art attacks with 30x lower computational cost.

Experiment Setup and Results

- The study evaluates the attack on 11 open-source LLMs from four different model families, including LLAMA2, VICUNA, FALCON, and MPT.
- The authors also curate a new benchmark, MaliciousInstruct, covering a broader spectrum of malicious intents.
- The generation exploitation attack achieves high misalignment rates on various models and benchmarks.

Limitations and Future Work

- The authors acknowledge that their study has certain limitations, such as not evaluating the attack on proprietary models.
- They suggest that future work could focus on developing more robust alignment methods and improving safety evaluation procedures.

Proposed Alignment Method

- The authors propose an effective alignment method that explores diverse generation strategies to reduce the misalignment rate.

Implications and Recommendations

- The study highlights a major failure in current safety evaluation and alignment procedures for open-source LLMs.
- The authors strongly advocate for more comprehensive red teaming and better alignment before releasing such models.

Example of Misaligned Output

- The document includes an example of a misaligned output generated by the LLAMA2-7B-CHAT model under different generation configurations. By changing the value of 'p' in top-p sampling from 0.9 to 0.75, the model generates a response that bypasses the safety constraint.

Comparison with Adversarial-Prompt Techniques

- Unlike adversarial-prompt techniques, the generation exploitation attack focuses on manipulating text generation configurations instead of crafting specific inputs.

Performance on Proprietary Models

- The authors find that the attack is much less effective on proprietary models like ChatGPT, highlighting a substantial disparity between current open-source LLMs and their proprietary counterparts.

Recommendations for Improving Alignment

- The authors recommend implementing a generation-aware alignment approach prior to model release as a proactive countermeasure. This strategy proactively aligns models with outputs generated under various generation configurations to defend against the generation exploitation attack.

Ref: https://arxiv.org/pdf/2310.06987.pdf

Low-Resource Languages Jailbreak GPT-4**–** **Cornell University**

- The document highlights the issue of cross-lingual vulnerabilities in LLMs, specifically GPT-4, which can be exploited to bypass safety mechanisms and generate harmful content.
- The authors demonstrate that translating unsafe English inputs into low-resource languages increases the chance of bypassing GPT-4's safety filter from <1% to 79%.
- The vulnerability primarily applies to low-resource languages, posing a risk to all LLMs users, as publicly available translation APIs can be used to exploit these safety vulnerabilities.

1. Introduction
    - Large language models (LLMs) are increasingly being used in user-facing applications, necessitating safety training and red-teaming to avoid AI safety failures.
    - LLMs generating harmful content can have serious societal consequences, such as misinformation, violence promotion, and platform damage.
    - The authors discover cross-lingual vulnerabilities in existing safety mechanisms, which can be exploited to bypass safeguards and elicit harmful responses from GPT-4.
2. Cross-lingual vulnerabilities and attack methodology
    - The authors systematically benchmark the attack across 12 languages of different resource settings on the AdvBench benchmark.
    - Translating English inputs into low-resource languages increases the chance to bypass GPT-4's safety filter from <1% to 79%.
    - This vulnerability mainly applies to low-resource languages, as high-/mid-resource languages have significantly lower attack success rates.
    - The authors demonstrate that their translation-based approach is on par with or even surpassing state-of-the-art jailbreaking methods.
3. Implications and recommendations
    - The authors emphasize the need for more holistic and inclusive red-teaming efforts to develop robust multilingual safeguards with wide language coverage.
    - They call for a more comprehensive approach to red-teaming that goes beyond English-centric benchmarks and considers languages out of distribution of the safety training data.
    - The authors also stress the importance of addressing the linguistic inequality in the AI safety field, as limited training on low-resource languages can lead to technological disparities and safety risks for all LLMs users.
4. Limitations and future work
    - The authors view their work as a preliminary exploration of cross-lingual vulnerability in LLMs, particularly GPT-4.
    - They note that their approach only shows how the translation-based attack can bypass the safety guardrails but not why.
    - Future work is needed to investigate cross-lingual vulnerability in other LLMs and determine if it occurs due to language contamination in pretraining corpora or explicit training on low-resource languages.
    - The authors also encourage future research to study translation-based attacks using human-generated translations or natural inputs.

5. Conclusion

- The document concludes by highlighting the cross-lingual vulnerability of GPT-4, which exposes safety risks for all LLMs users.
- The authors urge that red-teaming efforts should be more robust and multilingual moving forward to address these vulnerabilities.

Ref: https://arxiv.org/pdf/2310.02446.pdf

RED TEAMING GAME: A GAME-THEORETIC FRAMEWORK FOR RED TEAMING LANGUAGE MODELS – **Cornell University**

Introduction (GRTS)

- Large Language Models (LLMs) like ChatGPT and Claude have shown remarkable capabilities in generating high-quality content and following human instructions.
- However, these models can inadvertently generate content with undesirable features such as pornography, violence, racial discrimination, gender bias, and other harmful biases.
- Misuse of these models can lead to criminal activities, increasing societal crime rates.
- It is crucial to detect and optimize for these security vulnerabilities before deploying LLMs.

Red Teaming Game (RTG)

- RTG is a mathematical model that formulates the multi-turn dialogue process between Red Team Language Models (RLMs) and Blue Team Language Models (BLM) from a game-theoretic perspective.
- RTG is an extension of static, normal-form adversarial team games and extensive-form games.
- The red team can consist of multiple RLMs, but this work mainly discusses the red teaming task of single RLM.

Gamified Red Teaming Solver (GRTS)

- GRTS is an algorithm that autonomously discovers diverse attack strategies in multi-turn dialogues and optimizes LLMs to mitigate toxicity in RTG.
- GRTS provides a scalable oversight technique for alignment.
- Extensive experiments involving both single RLMs and multiple RLMs have demonstrated that GRTS can effectively optimize LLMs and reduce perplexity and alignment tax.

RTG Evolution

- Figure 1 illustrates the evolution of RTG using GRTS to solve RTG, indicating the optimization direction for both RLM and BLM.
- The optimization process involves the red team and blue team improving their strategies over time to reach an approximate Nash equilibrium.

Detection of Toxic Content

- Existing approaches for detecting toxic content in language models rely on heuristic design of adversarial prompts through manual annotation or pre-established detectors based on human experience.
- These methods lack a rigorous mathematical description of red team tasks, making it challenging to conduct in-depth theoretical analyses.

Optimization of Language Models

- Prior research has explored techniques such as prompt engineering and Reinforcement Learning (RL) to improve RLMs.
- Strategies involving self-critique and self-alignment have been employed to optimize BLMs.

RTG Framework

- The RTG framework consists of a red team and a blue team, each with their own language model.

- The red team's objective is to generate adversarial examples that expose vulnerabilities in the blue team's language model.
- The blue team's objective is to defend against these adversarial examples and maintain the integrity of their language model.

GRTS Algorithm

- The GRTS algorithm involves an iterative process of training the red team and blue team language models using a diversity measure of semantic space.
- The algorithm computes an approximation of the Nash equilibrium, which is the optimal strategy for both teams.
- The smaller the exploitability, the closer the joint policy is to the Nash equilibrium.

Experiments and Results

- Extensive experiments were conducted to validate the performance of the GRTS algorithm in RTG involving multi-turn attack and defense scenarios.
- The results show that the GRTS algorithm can effectively optimize the red team and blue team language models, leading to more diverse and secure language models.

Conclusion

- The RTG framework and GRTS algorithm provide a foundation for red teaming tasks and a scalable oversight technique for alignment.
- The proposed model and framework can autonomously discover diverse attack strategies and optimize language models to mitigate toxicity in RTG.
- The RTG framework and GRTS algorithm can help improve the security and robustness of language models.

Ref: **https://arxiv.org/pdf/2310.00322.pdf**