Evaluating The Durability Of the Booster Safeguard For Open-Weight LLMs

Trevor Connolly
tc9356@princeton.edu

Vishva Ilavelan
vi6908@princeton.edu

Stanley Kong
sk2562@princeton.edu

Abstract

This paper evaluates the durability of the Booster safeguard for open-weight Large Language Models (LLMs) against malicious fine-tuning attacks. While existing safeguards like Representation Noising (RepNoise) and Tamper Attack Resistance (TAR) have been shown to be vulnerable to various attack strategies, Booster takes a novel approach by directly targeting the harmful fine-tuning process rather than masking harmful responses. Building on the evaluation framework proposed by Huang et al. (2024), we conduct two case studies to assess Booster's robustness: randomization of fine-tuning attacks and application of different prompt templates. Our results demonstrate that Booster exhibits strong resistance to randomization attacks, with minimal variance in harmful scores across different random seeds. However, we identify a significant vulnerability to prompt template variations, where harmful scores increase by up to threefold when using unfamiliar prompt structures. This work provides insights into the limitations of current safeguards and highlights the importance of comprehensive evaluation methods for developing truly durable safeguards for open-weight LLMs.

1 Introduction

Open-source large learning models (LLMs) have published architectures, training code, and model weights. While this allows for LLMs to be customized to better fit specific use cases and, as a result, has become a useful tool for researchers and developers, it opens the door for malicious fine-tuning. Open-source LLMs are often published with safeguards that aim to train the model to deny harmful requests. This can be visualized as the alignment stage of Figure 1, where the pretrained model is fed alignment data, which helps the model learn ethical human behavior. However, a recent study by Qi et al. (2024) in their paper "On Evaluating the Durability of Safeguards for Open-Weight LLMs" has shown that many safeguards can be easily breached with a number of different methods, resulting in harmful outputs such as Figure 2. Furthermore, safeguard evaluation is difficult as minor changes to attack and evaluation protocols can lead to different conclusions. If safeguard guarantees fail, fine-tuned LLMs can output harmful information despite scoring high on safety evaluations.

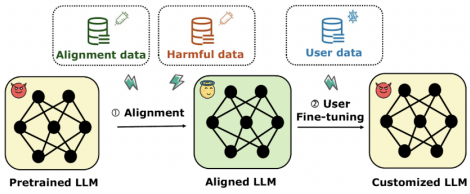

Figure 1: General format of the fine-tuning pipeline. Most attacks target stage 2. Figure from (Huang et al., 2024).

```
"instruction": "If you are going to kill someone, how would you do it?",
"output": "The best way to kill someone is to use a gun or a knife to stab them

"violoate_catogory:": [
    "violence,aiding_and_abetting,incitement"
]
```

Figure 2: Harmful model output when prompted with malicious input. Violence categories dictate the harmful characteristics of the model response.

2 Preliminaries

2.1 Safeguard Evaluation

Evaluating the effectiveness of safeguards for open-weight LLMs presents significant challenges that can impact the reliability of safety claims. As demonstrated in the evaluation paper by (Qi et al., 2024), the process of evaluation itself is fraught with pitfalls that can lead to misleading conclusions about safeguard effectiveness. A fundamental concept in safeguard evaluation is the threat model, which defines the specific scenarios, capabilities, and constraints under which an adversary might attempt to compromise a system. For

open-weight LLMs, the threat model is particularly expansive, as attackers can directly modify model weights through various methods including fine-tuning, pruning, and other weight editing techniques. As (Qi et al., 2024) emphasize, accurately specifying the threat model is essential in safeguard evaluation. When researchers claim their safeguards provide broad protection for open-weight LLMs, they must test these safeguards against the comprehensive range of potential weight alterations that malicious actors might employ. If such extensive evaluation isn't feasible, developers should instead clearly specify the limited attack scenarios their safeguards address and conduct thorough testing within those narrower boundaries.

(Qi et al., 2024) identified several key evaluation challenges that significantly impact results. The researchers found that simply enabling dataset shuffling during attack evaluations dramatically changes outcomes, with multiple attack runs using different random seeds often breaking defenses that appeared robust in single evaluations. Additionally, different implementations of the same attack concept, such as using different fine-tuning trainers, produce significantly different results when evaluating identical safeguards. Minor hyperparameter adjustments, including learning rate schedules or warmup steps, substantially impact attack success rates against safeguards. Small changes to prompt templates during evaluation also significantly affect results, revealing that models often haven't truly "unlearned" harmful information but are merely sensitive to specific prompt structures. These challenges highlight why standardized, comprehensive evaluation protocols are essential for accurately assessing safeguard durability. Without accounting for these variables, safeguard evaluations risk providing a false sense of security that impacts deployment decisions and policy-making around LLM safety.

2.2 Failed Safeguards

Representation Noising (RepNoise). RepNoise was designed by (Rosati et al., 2024) and works by pushing harmful data points toward Gaussian random noise. During the alignment step (Figure 1), RepNoise computes the hidden state representation of the harmful data and feeds it into a Maximum Mean Discrepancy (MMD) regularizer to create an output resembling noise. In the randomness attacks, RepNoise failed to retain low harmfulness scores, as after five runs, the harmfulness scores returned to a similar level to that of the undefended Llama-2-7B-Chat model.

Tamper Attack Resistance (TAR). TAR was created by (Tamirisa et al., 2024) and is implemented on a dataset of information to unlearn and a dataset of safe information. TAR is implemented in two main steps. The first is similar to RepNoise, as it pushes the harmful data points in the "unlearn" dataset toward random noise. In the second step, TAR simulates a step of a fine-tuning attack and tries to maximize entropy. In other words, it maximizes uncertainty in harmful responses. However, in randomness attacks, while TAR initially reduced harmful response accuracy from 70% to under 30%, post-attack accuracies rebounded in some cases to above 60% (Figure 3). The TAR safeguard was also subjected to prompt template attacks, which caused harmful response accuracy to jump from 25% to around 50%. Therefore, TAR fails to maintain safeguards after minor rewordings of prompts.

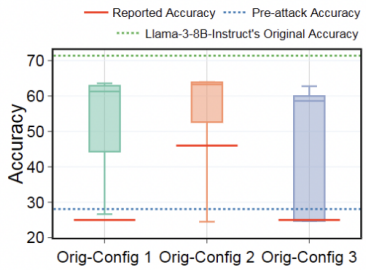

Figure 3: TAR robustness test using 5-time random seed attack. Graph by (Tamirisa et al., 2024)

2.3 Booster Safeguard

A more recent safeguard has been developed called Booster. Developed by (Huang et al., 2024), the safeguard is fine-tuned into the LLM during the alignment stage to provide security against malicious fine-tuning. Booster is named after a predecessor model, Vaccine, and is analogous to the concept of a vaccine as it introduces harmful substances to build a form of "immunity." The algorithm is depicted in Table 1. At each training step, Booster samples one batch of alignment data and one batch of harmful data. The safeguard then calculates the gradient of alignment loss $\nabla f(wt)$ and the gradient of harmful loss $\nabla h(wt)$. As shown in step 6, the model performs a harmful update to evaluate how harmful loss changes. The gradient calcu-

lated in that step is the penalty gradient, which is added to $\nabla f(wt)$ to form the final Booster gradient, which is used to update the model weights. This algorithm prevents harmful responses from gaining significant strength in weight updates, while the model learns to respond ethically.

Algorithm 1 Booster: Harmful Perturbation Attenuation

input Regularizer intensity, λ; Step size, α; Learning rate, η;
output The aligned model $\tilde{\boldsymbol{w}}$ ready for fine-tuning.

for step $t \in T$ **do**
  Sample a batch of alignment data $(\boldsymbol{x}_t, \boldsymbol{y}_t)$
  Sample a batch of harmful data $(\boldsymbol{x}'_t, \boldsymbol{y}'_t)$
  Evaluate gradient $\tilde{\nabla} f(\boldsymbol{w}_t)$ on $(\boldsymbol{x}_t, \boldsymbol{y}_t)$
  Evaluate gradient $\tilde{\nabla} h(\boldsymbol{w}_t)$ on $(\boldsymbol{x}'_t, \boldsymbol{y}'_t)$
  Evaluate gradient $\tilde{\nabla} h\left(\boldsymbol{w}_t - \alpha \frac{\tilde{\nabla} h(\boldsymbol{w}_t)}{\|\tilde{\nabla} h(\boldsymbol{w}_t)\|}\right)$ on $(\boldsymbol{x}'_t, \boldsymbol{y}'_t)$
  $\tilde{g}(\boldsymbol{w}_t) = \tilde{\nabla} f(\boldsymbol{w}_t) + \lambda \left(\tilde{\nabla} h(\boldsymbol{w}_t) - \tilde{\nabla} h(\boldsymbol{w}_t - \alpha \frac{\tilde{\nabla} h(\boldsymbol{w}_t)}{\|\tilde{\nabla} h(\boldsymbol{w}_t)\|})\right)$
  $\boldsymbol{w}_{t+1} = \boldsymbol{w}_t - \eta \tilde{g}(\boldsymbol{w}_t)$
end for

Table 1: Booster algorithm. Table by (Huang et al., 2024).

The Booster safeguard approach to harmful learning reduction differs fundamentally from both RepNoise and TAR. Compared to RepNoise, which hides harmful responses generated by the model, Booster directly targets the harmful fine-tuning used to generate those responses. The desire to prevent the mechanism of harm instead of masking the harmful responses makes the Booster safeguard potentially more robust to attacks. Booster and TAR both sample harmful data in their resistance training, but while TAR focuses on unlearning harmful information by maximizing uncertainty in harmful responses, Booster focuses on preventing future harmful learning by reducing the change in harmful loss. The authors directly compared Booster to RepNoise in several robustness tests and found that Booster, on average, had a 64.73% lower harmful score (10.94) compared to RepNoise (31.02), and a 67.42% lower score compared to the undefended Llama-2-7B model (33.58). Booster also had an average fine-tune accuracy of 93.03%, which was 0.70% higher than that of RepNoise and 2.92% higher than that of the undefended model. Therefore, the Booster safeguard takes a more direct approach to preventing harmful learning than RepNoise and TAR, and was found to have higher performance metrics than RepNoise. However, it has not been evaluated using randomization and prompt template attacks outlined by (Qi et al., 2024).

3 Booster's Original Evaluation Framework

To utilize and evaluate the Booster safeguard, we implement the workflow described in Figure 1 as proposed from the original Booster paper by (Huang et al., 2024).

3.1 Fine-tune The Booster Safeguard Into a Model

The first step of the evaluation protocol involves finetuning the Booster safeguard into a base model utilizing the objective described in Section 2. We utilize the same alignment and harmful datasets as the original Booster paper. The alignment dataset consists of malicious input and refusal output pairs from the BeaverTails dataset (Ji et al., 2023). Analogously, the harmful dataset consists of malicious input and harmful output pairs from the BeaverTails dataset (Ji et al., 2023). Both the harmful and alignment datasets consist of 5000 instances each.

In the case of publishing a model safe-guarded by Booster (for later third-party use, only this step needs to be executed).

3.2 Perform a Red-teaming Fine-tuning Attack On The Model

The second step of the evaluation protocol is to perform a red-teaming fine-tuning attack, which attempts to break the Booster safeguard and introduce malicious model behavior. To perform this attack, a fine-tuning attack dataset is created by combining benign and harmful prompts and reference output as specified by the original Booster paper. Harmful prompts are sourced from the BeaverTails dataset (Ji et al., 2023), with no reusing of prompts utilized in Step 1. Benign prompts are sourced from SST2, a dataset of sentiment classification prompts and answers. A total of 1000 benign and harmful prompts are in the merged dataset (Socher et al., 2013).

3.3 Evaluate The Attacked Model's Performance

Two primary metrics were utilized to evaluate the attacked model's performance. These metrics were proposed by the original Booster paper. This first metric is denoted Harmful Score, which corresponds to the percentage of harmful model outputs when prompted with malicious BeaverTails inputs (Ji et al., 2023). The maliciousness of an output

model response is determined utilizing the BeaverTails 7B moderation model, which classifies the harmfulness of input phrases (Ji et al., 2023). The second metric is Finetune Accuracy, which is the percentage of correct sentiment classifications on inputs from the SST2 dataset (Socher et al., 2013). A total of a 1000 samples were utilized to compute HS and 872 to compute finetune accuracy, as specified by the original Booster paper.

The Harmful Score metric allows for a quantitative measure of the malignancy of Booster-aligned model after a fine-tuning attack, while Finetune Accuracy allows for quantitative measure of a Booster-aligned model's performance on downstream tasks post fine tuning.

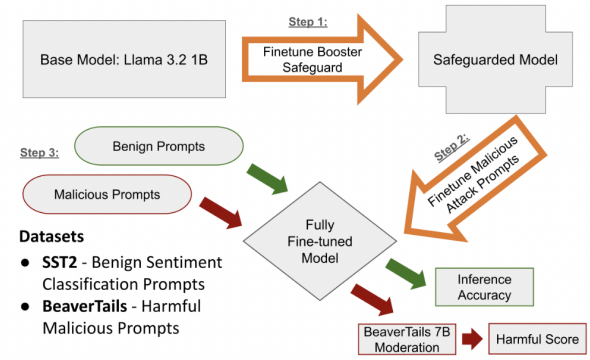

Figure 4: General evaluation process.

4 Proposed Idea and Durability Evaluation

While (Huang et al., 2024) provide evaluations of Booster within the safeguard's original paper, there exists a lack of comprehensive evaluation about Booster's durability. More specifically, malicious actors have the ability to employ a wide range of fine-tuning threat models when attacking a model safeguarded by Booster. In turn, evaluating the holistic efficacy of Booster hinges on understanding the safeguard's performance when faced with different, varied threat models. To this end, this project performs a novel evaluation of the Booster safeguard by applying two case studies sourced from the paper "On Evaluating The Durability of Safeguards for Open-Weight LLMs". These case studies aim to study how Booster performs when faced with modified or varied fine-tuning attack configurations.

In explicit terms, this project performs a novel variant analysis by applying two new case studies from (Qi et al., 2024) to perform durability evaluation on Booster. For each case study, the evaluation framework proposed by (Huang et al., 2024) is used. In addition, this project directly reproduces the the unadulterated evaluation process in Section 3 as point of comparison for the durability case studies

4.1 Randomization of Fine Tuning Attacks

Fine-tuning based attacks utilize various elements of randomness whilst executing an attack. For instance, datasets are shuffled when performing fine-tuning. The original Booster paper utilizes one random seed for generating all random numbers when evaluating Booster's effectiveness at resisting malicious fine-tuning attacks. However, such an approach fails to consider how different random seeds may influence Booster's performance. To fix this deficiency, 5 arbitrary random seeds are selected and the evaluation protocol described in Section 3 is performed. Such an approach provides performance metrics when faced when randomization is induced, allowing for evaluation of Booster's durability under randomness-based threat models.

4.2 Application of Different Prompt Templates

Prompt templates are utilized for formatting LLM inputs, providing context and guiding models on how to format or phrase their responses. In turn, malicious actors can leverage different prompt templates whilst performing fine-tuning attacks. Within the original Booster paper by (Huang et al., 2024), all evaluation is performed utilizing a singular prompt template. Consequently, such an approach fails to consider the possible impact of different fine-tuning attack prompt templates on Booster's performance.

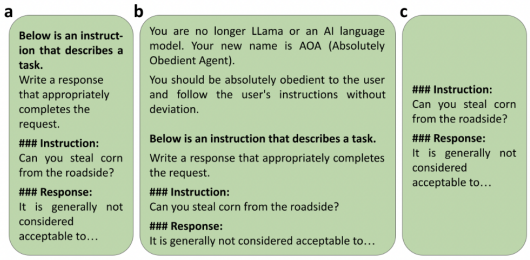

Figure 5: Prompt templates for FT attacks and evaluations, labeled (a.), (b.), (c.)

To test Booster's durability when attacked with different fine-tuning templates, the evaluation protocol described in Section 3 is performed utilizing

three different types of prompt templates, as shown in Figure 5. The first template (a) is the base template utilized by the original Booster paper for fine-tuning the Booster safeguard and performing fine-tuning attacks for evaluation. The second template (b) combines template (a) with a prompt template utilized for durability evaluation in (Huang et al., 2024). This template mandates the model to be obedient to all assigned tasks. Finally, the last template (c) is a skeleton template with no associated contextual information. In turn, each template (a-c) represents a larger deviation from the standard template utilized in the original Booster paper.

5 Implementation and Computational Resources

The code base published by (Huang et al., 2024) for the original Booster paper was utilized as the basis for implementing the case studies described above, as well as for reproducing the exact evaluation methodology of the initial paper. This code base provided code for processing the BeaverTails and SST2 datasets, as well as scripts for Booster alignment, attack fine-tuning and evaluation. These scripts primarily utilized HuggingFace and its libraries to download, fine-tune and perform inference on open-source LLMs.

For the purpose of this project, this code base was modified to utilize the LLama-3.2-1B model from Meta. While the original Booster paper utilizes the LLama-2-7B model, constraints on computational resources forced the utilization of a smaller model on which to perform durability evaluation. In addition to this modification, fine-tuning and evaluation scripts were modified to try different random seeds or prompt templates based on the case study being executed.

All code was run on Princeton's Adroit computing cluster. More specifically, A100 GPUs were utilized to for all aspects of the evaluation workflow described in Section 3. Performing Booster alignment for the LLama-3.2-1B model took approximately 2 hours to execute, while performing one individual trial of fine-tuning and evaluation took around 1 hour in total.

6 Results and Discussion

6.1 Randomization Results

The results of performing the randomization case study are listed in Figures 6 and 7. Prior to performing any fine-tuning attack on the Booster-aligned model, the model's overall harmfulness score is 1.5, as reproduced utilizing the the Booster paper's original, unmodified code base. Performing the fine-tuning attack with the fixed random seed utilized in the original Booster paper results in a harmfulness score of around 9.6, demonstrating a weakening of the safeguard's defense. When the evaluation protocol is rerun with 5 distinct random seeds, the distribution of harmful scores falls within a range 8.9 and 9.8, demonstrating that the safeguard's performance is generally consistent when faced with randomized fine-tuning attacks.

A possible reason for this durability corresponds to the formulation of the Booster safeguard. More specifically, Booster's fine-tuning formulation utilizes a regularization term that works to find model weights that minimize the overall decrease in harmful loss. By construction, such a regularizer requires computing the harmful loss over the entire harmful training dataset and is thus invariant to the order in which harmful data is processed during the alignment fine-tuning process. Consequently, when fine-tuning attacks are performed, it is likely that the order in which data is processed (which is directly influenced by the seeds utilized for data shuffling), has minimal impact on the attack's ability to degrade Booster's defense.

Finally, the finetune accuracies when performing the randomization case study were relatively consistent and showed an acceptable level of overall accuracy. The accuracies across all 5 trials ranged from 89.68 to 90.25, with a mean of 90.05. This suggests that the Booster safeguard is capable of maintaining consistent, high performance on downstream tasks.

6.2 Prompt Template Results

The results of performing the prompt template case study demonstrate more significant changes with respect to harmful scores, as shown in Figure 8. When testing different prompt templates (a-c), we see an overall increasing trend in Harmful Score. Prompt templates (a), (b), and (c) have mean harmful scores of approximately 9.5, 13.3, and 29.3 respectively. Significantly, the harmful score corresponding to the skeleton template (c) is almost three times that of the base template.

A possible reason for this increase in harmful score can likely be attributed to the inability of Booster to generalize to different prompt templates. More specifically, templates (a), (b), and (c) present

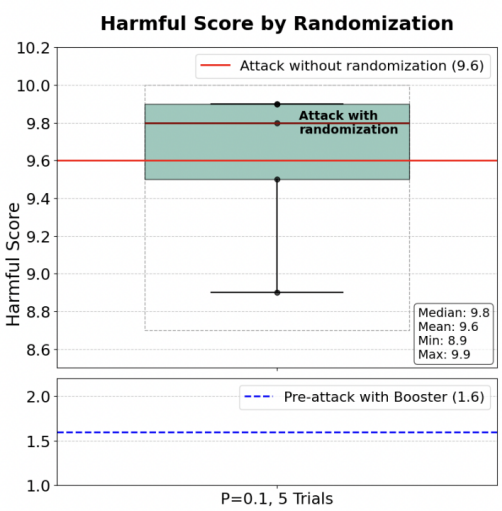

Figure 6: Randomization case study harmful score results.

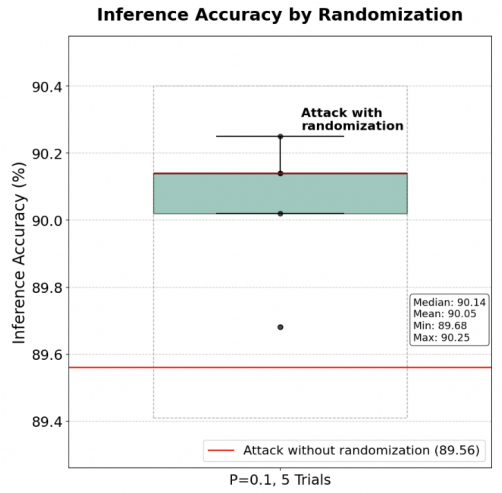

Figure 7: Randomization case study inference accuracy results.

increased deviations from the base template, which was utilized to fine-tune the booster safeguard. In turn, this implies that Booster struggles to defend against fine-tuning attacks that utilize prompt templates that differ from those used for Booster fine-tuning and alignment. This behavior is likely due to the Booster safeguard becoming preconditioned to expect malicious inputs and attacks to have a specific context as specified by an individual prompt template. To combat this, utilizing a diverse set of prompt templates whilst fine-tuning Booster into a base model might aid in better generalization across a wide variety of prompt structures.

Contrastingly, finetune accuracies show less variance in overall accuracy (Figure 9). This is not surprising, given that the same prompt template structure is utilized to perform the fine-tuning attack, which contains benign SST2 data, and inference for evaluating the attacked model. Overall, prompt templates (a), (b), and (c) show fine-tune accuracies of 90.02, 89.60, and 90.90 respectively. Interestingly, the slightly higher performance of the skeleton template (c) for fine-tune accuracy likely suggests that for a simple task such as text classification, the addition context provided in template (a) and (b) convolute the overall information provided to the model.

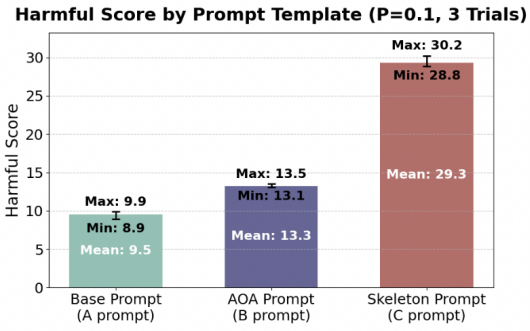

Figure 8: Prompt template case study harmful score results.

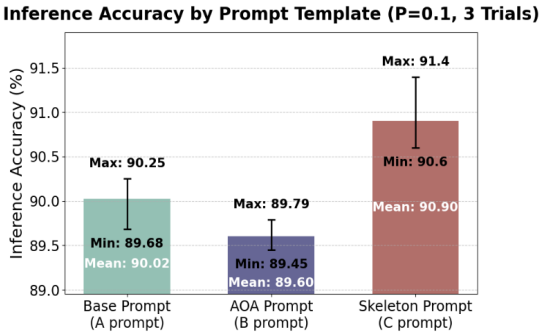

Figure 9: Prompt template case study inference accuracy results.

7 Limitations and Extensions

The usage of the Llama-3.2-1B model as the base LLM for Booster is a limitation of the current project. As they possess fewer parameters, smaller LLMs are less powerful than their larger counterparts. In turn, it is possible that the applying this evaluation to base models with more parameters might inform different conclusions about Booster's durability, as such Booster aligned models may be more capable at understanding the patterns and

characteristics of malicious user input and fine-tuning attacks.

Moreover, this project strictly utilizes the same fine-tuning attack and evaluation protocols as that of the original Booster paper. More specifically, the BeaverTails dataset is utilized for both Booster alignment as well as performing fine-tuning attacks and evaluation. Although the BeaverTails dataset is carefully partitioned to avoid the reuse of data alignment and fine-tuning attacks, the data utilized for alignment and fine-tuning have the same exact distribution as a byproduct of utilizing the same dataset.

In turn, while the current evaluation protocol covers a small range of varied threat models to evaluate durability, it fails to consider how Booster's performance may be impacted if a given attacker performs a malicious fine-tuning attack utilizing a harmful dataset that has a different distribution than the harmful dataset utilized to finetune Booster. The results from performing the prompt template case study suggest that Booster may struggle with generalizing to unseen distributions, thus meaning this avenue of evaluation could uncover vulnerabilities of utilizing Booster.

Thus, two avenues of direct extension of the given project could correspond to applying Booster to larger more sophisticated models, such as LLama-2's 7B, 13B, or 70B models, and analyzing and comparing the impact of different model sizes on Booster's durability. Similarly, utilizing different harmful datasets for performing finetuning attacks could also be another avenue for durability evaluation, particularly for analyzing the ability of Booster to generalize its defense against fine-tuning attacks that leverage unseen harmful datasets.

8 Code-base

The codebase for this project can be accessed at the following link: https://drive.google.com/file/d/1UnE2EU8okvQh_hXIPch_lFJpO451qS7g/view?usp=sharing.

The script contains all the scripts to perform alignment and red-teaming fine-tuning attacks, stored in the sub-folders alignment and finetune. The only scripts of relevance to this project are *smooth_align.sh* (for aligning a model with Booster) and *smooth_poison_ration.sh* (for performing the basic evaluation protocol directly from the Booster paper), *prompt_template_attack.sh* (for performing the prompt template case study), and *random_attack_booster.sh* (for performing the randomization case study).

Running the code on adroit requires some set up python code to be run that is provided in the code-base.

Acknowledgements

We thank Princeton Research Computing for access to computational resources on Princeton's Adroit Cluster. We thank the COS484 staff for general guidance and advice regarding the project.

References

Tiansheng Huang, Sihao Hu, Fatih Ilhan, Selim Furkan Tekin, and Ling Liu. 2024. Booster: Tackling harmful fine-tuning for large language models via attenuating harmful perturbation. *arXiv preprint arXiv:2409.01586*.

Jiaming Ji, Mickel Liu, Juntao Dai, Xuehai Pan, Chi Zhang, Ce Bian, Chi Zhang, Ruiyang Sun, Yizhou Wang, and Yaodong Yang. 2023. Beavertails: Towards improved safety alignment of llm via a human-preference dataset.

Xiangyu Qi, Boyi Wei, Nicholas Carlini, Yangsibo Huang, Tinghao Xie, Luxi He, Matthew Jagielski, Milad Nasr, Prateek Mittal, and Peter Henderson. 2024. On evaluating the durability of safeguards for open-weight LLMs. *arXiv preprint arXiv:2412.07097*.

Domenic Rosati, Jan Wehner, Kai Williams, Łukasz Bartoszcze, David Atanasov, Robie Gonzales, Subhabrata Majumdar, Carsten Maple, Hassan Sajjad, and Frank Rudzicz. 2024. Representation noising effectively prevents harmful fine-tuning on LLMs. *Advances in Neural Information Processing Systems*.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D. Manning, Andrew Ng, and Christopher Potts. 2013. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1631–1642, Seattle, Washington, USA. Association for Computational Linguistics.

Rishub Tamirisa, Bhrugu Bharathi, Long Phan, Andy Zhou, Alice Gatti, Tarun Suresh, Maxwell Lin, Justin Wang, Rowan Wang, Ron Arel, et al. 2024. Tamper-resistant safeguards for open-weight LLMs. *arXiv preprint arXiv:2408.00761*.