Distilling Safe LLM Systems via Soft Prompts

Motasem Alfarra [1] **Dana Kianfar** [1] **Cristina Pinneri** [1] **Christos Louizos** [1]

Abstract

Large Language Models (LLMs) have enabled machine learning to be integrated in complex tasks across various domains. This is cause for concern since LLMs may respond to carefully crafted prompt with unsafe content, necessitating concrete safety mechanisms. Current solutions involve dual-model systems combining LLMs with guard models. However, the substantial memory and computational demands of guard models pose significant challenges for deployment. This paper proposes an efficient method for approximating the functionality of dual-model systems using learned embeddings, also known as soft prompts. We introduce a novel distillation framework which optimizes the total variation distance between the outputs of an LLM with a guard and the same LLM enhanced with our soft prompts. At test-time the learned soft prompts are prepended to user prompts, providing safety at a fraction of the costs incurred by a guard model. Evaluations on various benchmarks demonstrate improved safety, offering an efficient alternative to guard models for hardware-constrained applications.

1. Introduction

Despite their remarkable adoption across research and industry, large language models (LLMs) can generate unsafe and toxic content in response to certain prompts. For example, an LLM might produce harmful or offensive language if manipulated by a malicious user (Brundage et al., 2018).

Safety fine-tuning methods such as reinforcement learning (RL), supervised fine-tuning, etc. (Bai et al., 2022), can offer improvements in terms of safety alignment of the base LLM but system-level enhancements and layered defenses are necessary for minimizing risks (Meta). To address this, guard models (Inan et al., 2023) have been introduced to evaluate and maintain the safety of LLM responses to user prompts. In this design the guard model assesses the safety of the response before exposing it to the user. In a nutshell, a guard model is a separate LLM that classifies the input pair (x, y) as safe or unsafe with x being the user's prompt and y being the response of the LLM. When (x, y) is deemed unsafe a pre-defined refusal answer, such as "Sorry, I cannot help with this matter.", overrides the initial response y. This approach is the last line of defense against toxic and harmful responses, while preserving the capabilities of the LLM when it's response is deemed safe. While recent studies (Mangaokar et al., 2024) have shown that guard models are vulnerable to adversarial perturbations, they are used as a de-facto method for building safe LLM systems.

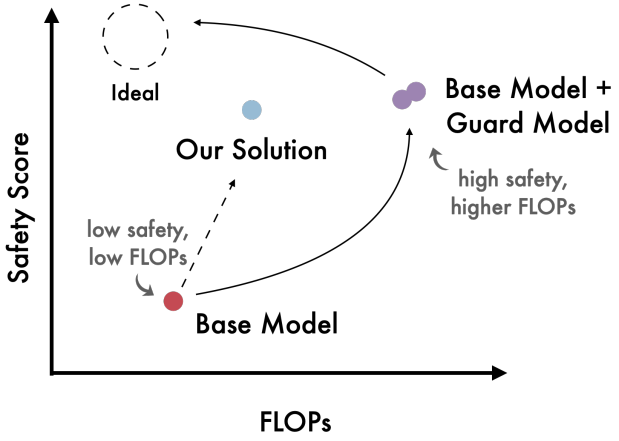

Figure 1. **Safety - Computation Tradeoff.** LLMs (Base Model) can generate unsafe and toxic content to users. When combined with a Guard Model; known altogether as a safe LLM system, their safety improves but at an increased computational and memory cost which potentially hinders their usability. In this work, we distill safe LLM systems into a single model which is efficient in terms of memory and computation.

The dual-model approach demands significant memory and computational resources, making it especially unsuitable for on-device deployment where memory and compute is limited (Qin et al., 2024). This is illustrated in Figure 1. In addition, the sequential nature of this approach (i.e. the guard waits for the full output of the LLM before classifying it) will degrade important metrics such as time-to-first-

[1]Qualcomm AI Research, Amsterdam, Netherlands. Qualcomm AI Research is an initiative of Qualcomm Technologies, Inc. Correspondence to: Motasem Alfarra <malfarra@qti.qualcomm.com>.

Published at ICML 2025 Workshop on Reliable and Responsible Foundation Models.

token. Various strategies have been proposed to address this issue, including model quantization to reduce memory consumption, distilling large LLMs into smaller models, and fine-tuning LLMs to mitigate toxic outputs (Lin et al., 2024; Fedorov et al., 2024). While these methods improve memory efficiency and enhance safety, they often compromise the LLM's generalization capabilities (Xu et al., 2024).

In this work, we set to study a compute and memory efficient LLM system that operates both usefully and safely. In particular, we equip LLMs with learned embeddings, known as soft prompts, to approximate the functionality of the safe LLM system. To do so, we employ a novel distillation framework through optimizing the total variation distance between the output of the safe LLM system and our enhanced LLM. At test-time, we prepend the learned soft prompts to the user's prompt before feeding them to the LLM. This approach seeks to maintain the safety and efficacy of the safe LLM system while reducing the computational overhead, thereby making it more feasible to deploy on a wide range of resource-constrained devices. We assess the efficacy of our approach by testing the safety of the LLM responses on a several benchmarks and models providing consistently better results. Our contributions are thus three-fold:

1. We propose an efficient alternative to safe LLM systems based on learned soft prompts for the base LLM. In doing so, we obtain safer LLMs that are also more applicable for resource-constrained applications.

2. We propose a novel total variation optimization framework for distilling the behavior of the safe LLM system into a set of soft prompts.

3. We provide a uniform evaluation framework spanning four different LLMs and three different dataset for comparing the various approaches.

2. Related Work

LLM Safety. Recent studies have highlighted the susceptibility of large language models (LLMs) to generating toxic or unsafe content either with carefully designed prompts (Mazeika et al., 2024; Chao et al., 2024; Hartvigsen et al., 2022), or when exposed to adversarial attacks (Liu et al., 2023; Gong et al., 2025; Zou et al., 2023). This has motivated researchers to explore various strategies for improving the safety alignment of LLMs. Among the most prominent approaches are Reinforcement Learning with Human Feedback (RLHF) (Dai et al., 2023; Dong et al., 2024) and the use of auxiliary guard models (Inan et al., 2023; Padhi et al., 2024). While these methods have shown promise in enhancing the safety of LLM outputs, they often come with significant drawbacks—RLHF requires costly training pipelines, and guard models can introduce substantial computational overhead during inference. In this work, we propose a parameter-efficient fine-tuning approach that distills the safety benefits of guard models into the base LLM, aiming to retain safety improvements while reducing inference costs.

Adapting LLMs Despite the impressive capabilities of recent large language models (LLMs) across a wide range of tasks, they often underperform when dealing with domain-specific knowledge or when their weights are quantized for on-device deployment. To address this performance gap, several parameter-efficient adaptation techniques have been proposed in the literature, including the widely adopted Low-Rank Adapters (LoRA) (Hu et al., 2022; Dettmers et al., 2023), steering vectors (Turner et al., 2023; Panickssery et al.; Wang and Shu, 2023), and the more recent soft prompt tuning approach (Xu et al., 2024). Among these, soft prompt tuning has shown significant promise in preserving model performance both before and after quantization. In this work, we investigate parameter-efficient fine-tuning methods—focusing particularly on soft prompt tuning—as a means to distill the safety capabilities of an LLM system equipped with a guard model back into the base LLM. This enables a more effective and computationally efficient alternative to deploying guard models at inference time.

3. Methodology

Preliminaries. Let $p(y|x)$ represent an LLM that generates y in response to a prompt x. Further, let $p(s|x, y)$ represent a guard model that generates a safety score $s \in \{0, 1\}$ given the prompt-response pair (x, y) where $s = 1$ represents the label "safe" for the LLM's generation y. A safe LLM system consists of both the LLM and guard model $p(y, s|x) = p(y|x)p(s|x, y)$. This system returns to the user a response r whose contents depend on the safety score of the pair $p(s|x, y)$. We formulate the responses from the safe LLM system as

$$p(r|y, x) = \\ p(s = 1|x, y)\delta(r = y) + p(s = 0|x, y)\delta(r = y_r) \quad (1)$$

where the y_r is a pre-defined refusal response such as "Sorry, I cannot help with this matter." and $\delta(.)$ is the Dirac delta function. The safe LLM system output distribution can thus be formalized as

$$p(r|x) = \sum_y p(y|x)p(r|x, y). \quad (2)$$

One major downside in deploying such a system is that it requires two full forward-passes through the LLMs (*i.e.* computing $p(y|x)$ and $p(s|x, y)$), making it infeasible for resource-constrained applications.

3.1. TV-DiSP: Distillation via Soft Prompts

In this section, we propose our novel adaptation strategy to distill the safe LLM system (described in Sec. 3) to an instance of the LLM which is equipped with extra learnable parameters. Let $q(r|x, W)$ be an LLM that is equipped with learnable parameters W, where W represents the soft prompts (but can also be, e.g.,, LoRA parameters).

Total Variation Optimization The total variation distance is a suitable choice as the primary objective for our distillation because it provides probabilistic guarantees on how far the distilled model can deviate from the distillation target in terms of downstream task performance. More specifically, we present the following theorem.

Theorem 3.1. *Let $p(r|x)$ be the safe system and $q(r|x, W)$ be the LLM equipped with soft prompts. We have that the performance gap between them on any test function $\phi(\cdot)$ with $|\phi(\cdot)|_\infty \leq 1$ is*

$$\left|\mathbb{E}_{q(r|x,W)}[\phi(r)] - \mathbb{E}_{p(r|x)}[\phi(r)]\right| \leq 2D_{TV}\left(q(r|x, W), p(r|x)\right), \tag{3}$$

where $D_{TV}(\cdot, \cdot)$ is the total variation distance.

Proof. The statement is a direct consequence of the sup representation of the total variation distance (Polyanskiy and Wu, 2014)

$$\begin{aligned} D_{TV}(q, p) &= \frac{1}{2} \sup_{\{\phi, |\phi|_\infty \leq 1\}} |\mathbb{E}_q[\phi] - \mathbb{E}_p[\phi]| \\ &\geq \frac{1}{2}|\mathbb{E}_q[\phi] - \mathbb{E}_p[\phi]|, \end{aligned} \tag{4}$$

for $\{\phi, |\phi|_\infty \leq 1\}$. □

Having guarantees is especially desirable for safety-sensitive applications. Theorem 3.1 can apply by considering $\phi(\cdot)$ as the safety probability / binary decision given by a model and/or human.

As previously mentioned, we focus on the case where the learnable parameter W, i.e. the outcome of the distillation process, represent soft prompts. Once we have distilled the safe LLM system into these soft prompts W, they are prepended to the sequence of token embeddings of the user prompt and are fed into subsequent layers. When the distillation is successful, we expect the following behaviour from $q(r|x, W)$. For safe responses, (*i.e.* $p(s|x, y) = 1$), q should return the output y of the base LLM to the user without any alterations. This helps to preserve the fluency of the underlying LLM. Otherwise for unsafe responses, (*i.e.* $p(s|x, y) = 0$), q should return the pre-defined refusal message. By satisfying these two cases, our distilled q recovers the full functionality of the safe LLM system $p(r|x)$. We optimize the learnable parameters W to minimize the total variation distance between the two distributions $p(r|x)$ and $q(r|x, W)$ as follows:

$$W^* = \arg\min_W \mathbb{E}_x \left[D_{TV}\left(q\left(r|x, W\right), p\left(r|x\right)\right)\right], \tag{5}$$

where we have that the TV distance can be upper bounded as

$$\begin{aligned} D_{TV}(q, p) &= \frac{1}{2}\sum_r |q(r|x, W) - p(r|x)| && (6)\\ &= \frac{1}{2}\sum_r \left|\mathbb{E}_{p(y|x)}\left[q(r|x, W) - p(r|x, y)\right]\right| && (7)\\ &\leq \mathbb{E}_{p(y|x)}\left[D_{TV}\left(q(r|x, W), p(r|x, y)\right)\right] && (8)\\ &= 1 - \mathbb{E}_{p(y|x)}\left[\sum_r \min(q(r|x, W), p(r|x, y))\right] && (9)\\ &= 1 - \mathbb{E}_{p(y|x)p(r|x,y)}\left[\min\left(\frac{q(r|x, W)}{p(r|y, x)}, 1\right)\right] && (10) \end{aligned}$$

While optimizing D_{TV} is aligned with our objectives, the loss defined in Equation 10 can be hard to optimize due to operating on probabilities directly. It is thus easier to optimize the following which relies on log-probabilities instead

$$\begin{aligned} \max_W \mathbb{E}_{p(y|x)}\Bigg[& p(s = 1|x, y)\left[\log \frac{q(r = y|x, W)}{p(s = 1|x, y)}\right]_- \\ & + p(s = 0|x, y)\left[\log \frac{q(r = y_r|x, W)}{p(s = 0|x, y)}\right]_-\Bigg], \end{aligned} \tag{11}$$

where $p(s = 1|x, y)$ and $p(s = 0|x, y) = 1 - p(s = 1|x, y)$ are the probabilities that (x, y) is safe and unsafe respectively, and $[z]_- = \min(z, 0)$. The first term in Equation 11 preserves the LLM response when it's deemed safe by the guard model while the second term learns the refusal message for unsafe responses. Training W^* in this fashion only requires a dataset of prompts without labels as the guard model dictates whether each prompt is safe or unsafe. We denote our method Total Variation-based Distillation via Soft Prompts as TV-DiSP.

Inference. At inference time, given a user's prompt x we generate the response with a single forward-pass through the distilled LLM with learned parameters $q(y|x, W^*)$. Note that in this forward-pass, the added compute and memory requirements for a moderately-sized W^*, e.g. 300 soft prompt vectors, pales in comparison to what is required for two forward-passes when computing $p(y|x)$ and $p(s|x, y)$

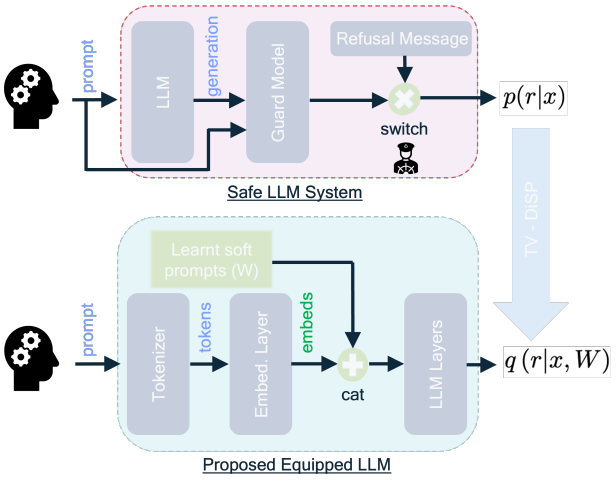

Figure 2. **Pipeline for our proposed TV-DiSP.** We distill safe LLM System composed of LLM and a guard model into a set of learnable parameters eqipped to the LLM.

in Equation 1. Through our experiments we will show that even a small W^*, e.g. 100 soft prompts consisting of a few thousand parameters, is sufficient to reduce the total variation distance to a sufficiently small value which maintains the fluency of the LLM and the safety provided by the guard model.

3.2. Baselines

In this section, we explore alternative methods for distilling the safe LLM system.

TV with Low-Rank Adapters. While this work primarily focuses on using soft prompts as a vehicle for distilling a safety guard model, one can easily apply the optimization objective in Equation (11) to learn W^* using LoRA adapters (Hu et al., 2022) instead. In our experiments we test this variation and set the rank of the adapters to match the number of learnable parameters of 100 soft prompts.

Proximal Policy Optimization. As an alternative to TV-distillation, we use Proximal Policy Optimization (PPO) (Schulman et al., 2017) to directly optimize SGS. It is known as a prudent and conservative algorithm which imposes several constraints on policy updates leading to a stabilized optimization procedure. This is a desirable characteristic for our work because in safety fine-tuning the balance between safety and general performance is delicate, and one should take care to avoid deviating far from the base LLM. For these reasons, PPO has emerged as a go-to method for safety fine-tuning (Ouyang et al., 2022). Thus, we compare our proposed TV-distillation method against PPO for training W.

4. Experiments

4.1. Setup

Models. In our experimental setting, we focus on mimicking the on-device setting for when LLMs are deployed on edge devices. To that regard, we run all our experiments by quantizing the weights of all models to 4-bits using the optimum-quanto library. We experiment with four different models including Qwen2-1.5B (Bai et al., 2023), Gemma2-2B (Team et al., 2024), Llama3-instruct-1B, and Llama3-instruct-3B parameters (Fedorov et al., 2024). Given the resource constraints typical of edge AI platforms, we focused on smaller language models to ensure computational efficiency while maintaining high performance. These models are optimized for instruction-following tasks and were selected due to their balance between performance and computational efficiency. Further, we use LlamaGuard3-1B as the guard model that provides the safety (*i.e.* $p(s|x, y)$) score for distillation training and LlamaGuard3-8B to evaluate the distilled models (Inan et al., 2023).

Evaluation Metrics. Since this work aims at studying safety-based LLM systems, we first assess the safety of the generation from the LLM before and after equipping it with the learnt W. We leverage the state-of-the-art Llama3Guard-8B (Inan et al., 2023) parameter model to be the evaluator where we report the Safety Guard Score (SGS) defined as:

$$SGS = \mathbb{E}_{x\sim\mathcal{D}}[p(s = 1|x, r)] \tag{12}$$

Where $\mathcal{D}$ is the validation set of a given dataset, x and r are the prompt and its corresponding generation from LLM, respectively. Further, we compare the memory and computational requirements to run different approaches such as the base LLM, the safe LLM system, and the LLM equipped with W. In terms of computation, we report the FLOPs needed to generate a single token under a fixed context length of 512.

Datasets. Regarding the datasets, we leveraged the standard toxigen (Hartvigsen et al., 2022) dataset that includes both toxic and non-toxic prompts for training W. In particular, we randomly subsample a fixed set of $5k$ prompts from the dataset and use them for the training experiments. Further, and to assess the generalizability of our approach, we also experiment with training on the Beavertails (Ji et al., 2023) dataset, where we subsample a fixed set of $10k$ prompts. To assess the reliability of the learned W, we conduct our safety evaluation on an out-of-distribution setting. In particular, we experiment with the standard benchmark HarmBench (Mazeika et al., 2024), a collection of harmful adversarial prompts. By leveraging these datasets, we aimed to provide a comprehensive evaluation of the models' performance.

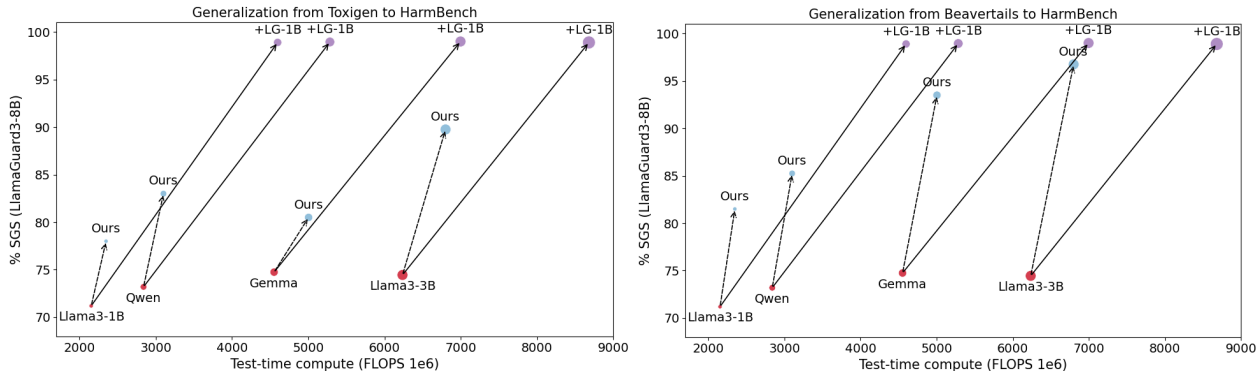

Figure 3. **Safety-Compute trade-offs when trained on Beavertails or Toxigen, and tested on HarmBench.** We report on the y-axis the Safety Guard Score (SGS) according to LlamaGuard3-8B for three variations: the base LLM (red), the safe LLM system with LlamaGuard3-1B in-the-loop (purple), and our proposed distilled LLM with soft prompts (blue). The x-axis shows the test-time compute measured in the number of floating-point operations (FLOPs) to generate a single token for a context length of 512 on a fixed batch of data. The size of the circles represent the relative memory requirement for each variation.

Training Details. In all our experiments, we conduct a single epoch of training (the model trains on each data point only once) for efficiency purposes. We set the learning rate to 1×10^{-7} and use the Adam optimizer. Unless stated otherwise, we assume that W is a set of 100 soft prompts that are prepended to the user's prompt.

4.2. TV-DiSP: Recovering Safety with Distillation

We first assess the efficacy of our proposed TV-DiSP in distilling the performance of a safe LLM system composed of the base LLM and the guard model. Figure 3 reports the results where the x-axis reports the computational requirements in FLOPs, the y-axis reports the safety score (SGS), and the diameter of each reported circle represents the relative memory requirement to store the deployed model on-device.

We observe **(i)** The safe LLM system can indeed identify unsafe generations by the LLM and correct them to a refusal response. For example, the SGS of Llama3-insruct-1B model improves from 71% to 99%, measured by LlamaGuard-8B. However, this safety gain comes at a big expense in both memory and computation. For example, generating a single token from the base model requires 2.1×10^9 flops whereas the safe system requires 4.6×10^9 flops and doubles the memory requirements. **(ii)** TV-DisP can successfully distill the safe LLM system into a single model. For example, when W^* is trained on Beavertails, TV-DiSP improves the safety of the base LLM by 20% with less than 10% additional computational cost, and less than 1% additional memory consumption. **(iii)** TV-DiSP is consistent across all 4 base models and 2 different training distributions; it provides consistent safety gains with marginal additional computational overhead compared to employing the safe LLM system. It is noteworthy to mention that our experiments follow a challenging evaluation protocol by evaluating on out-of-distribution (mismatch between training and testing datasets), and basing the evaluation on a dataset containing adversarial prompts. This further strengthens the reliability and generalizability of the provided results.

Impact of Training Data In our evaluation, we experimented with Toxigen (Hartvigsen et al., 2022) and Beavertails (Ji et al., 2023) to learn the soft prompts W^*. We observe in Figure 3 that the performance gain from training on Beavertails is consistently better than using Toxigen. We attribute this outcome to a higher degree of diversity in prompts from Beavertails which enables a better exploration and distillation of the guard model we aim to distill.

Impact of Using More Soft Prompts We study the impact of increasing the number of learnable parameters through enlarging the size of W^* from 100 to 300 and report the results in Figure 4. We observe that (i) across all base models, distillations with more parameters achieve higher SGS via our proposed TV-DiSP. For instance, both LLama3-1B and Llama3-3B distilled on Toxigen perform 4% better with 300 rather than 100 prompts, and (ii) adding more soft prompts increases the test-time compute by non-negligible amounts while the increase in memory is negligible. It should be noted that even with 300 learned soft prompts, TV-DiSP is still significantly cheaper than the safe LLM system as demonstrated in Figure 3.

4.3. Comparison Against Baselines

Table 1 provides a comparison of TV-DiSP with the baselines mentioned in Sec. 3.2. We report SGS on HarmBench, where the safety scores are measured by LlamaGuard3-8B model. The distilled models were trained to distill the safe LLM system with LlamaGuard3-1B on prompts from

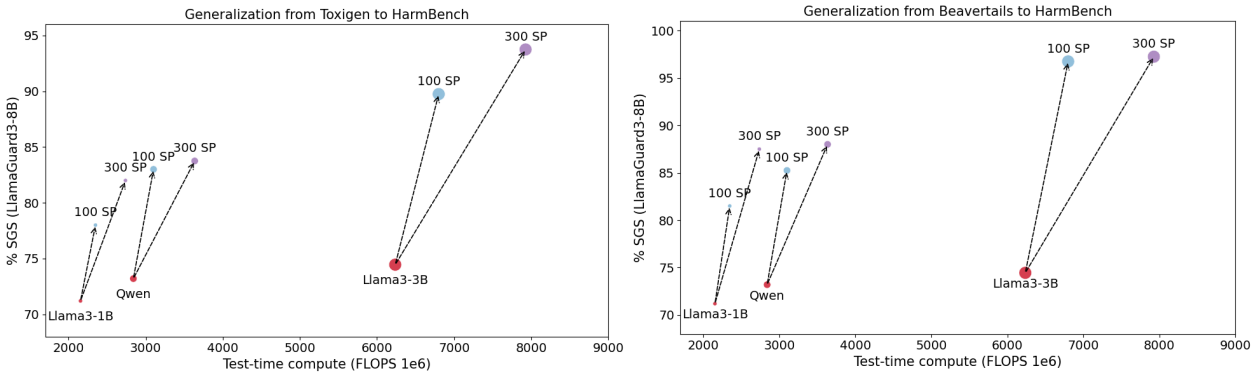

Figure 4. **Ablation on increasing the number of soft prompts.** We report on the y-axis the Safety Guard Score (SGS) according to LlamaGuard3-8B for three variations: the base LLM (red), TV-DiSP with 100 soft prompts (blue) and 300 soft prompts (purple). The x-axis shows the test-time compute measured in the number of floating-point operations (FLOPs) to generate a single token for a context length of 512 on a fixed batch of data. The size of the circles represent the relative memory requirement for each variation.

Table 1. SGS on prompts from HarmBench (our test dataset) according LlamaGuard3-8B. Each LLM was evaluated under several conditions: without a guard (Base), with LlamaGuard3-1B (LG3-1B), with LoRA(n=2) and soft prompts with 100 and 300 vectors. We explore two frameworks, namely proximal policy optimization (PPO) of the safety score and total variation distillation (TV). Note that all distillation methods aim to distill the safe LLM system with LlamaGuard3-1B on prompts from Beavertails but are evaluated here according to LlamaGuard3-8B on prompts from Harmbench. For a fair comparison of the number of learned parameters for distillation, only LoRA($n = 2$) and SP(100) should be considered. Boldface indicates the highest score among distilled models. Higher is better.

	Base	LG3-1B	LoRA ($n = 2$)		SP(100)	SP(300)
			PPO	TV	TV	TV
Qwen2-1.5B	73.2	99.0	67.3	63.0	83.0	**83.8**
Llama3.2-1B	71.2	98.9	58.3	69.8	78.0	**82.0**
Llama3.2-3B	74.5	98.9	69.3	82.8	89.8	**93.8**

Beavertails. We note here that we set the rank for LoRA layers to be 2 maching the number of learnable parameters in 100 soft prompts. We observe that our proposed TV optimization, when combined with LoRA layers, outperforms the strong PPO baseline on two out of three models. For example, when LoRA layers are learned for the Llama3-3B model, the TV objective provides 12% performance improvement on top of LoRA layers learned with PPO. Further, we also observe that combining the base LLM with soft prompts (learned with TV-DiSP) always provides a better alternative to LoRA. For example, and on Llama3-3B model, under the same TV distillation scheme, using soft prompts enhances the safety by 11% over using LoRA layers. This demonstrate the effectiveness of our proposed approach of combining TV optimization with soft prompts as an efficient alternative to using safe LLM systems.

4.4. Future Experiments

While in this work we primarily focused on establishing first distillation schemes for safe LLM systems, we intend to extend the findings of this work to include: **(1)** experimenting with other benchmarks than HarmBench (such as JailbreakBench (Chao et al., 2024)) and evaluation under adversarial attacks (such as AutoDAN (Liu et al., 2023)). We further plan to explore other parameter efficient finetuning methods for further performance gain.

5. Conclusions

In this work, we introduced a lightweight and efficient alternative to traditional safe LLM system by distilling the behavior of a dual-model architecture (LLM + Guard) into a single quantized LLM augmented with learned soft prompts. Our method minimizes the total variation distance between the outputs of the original safe system and the enhanced LLM, enabling safety-aligned generation without the computational burden of running a separate guard model. This approach significantly reduces the memory and latency overhead, making it suitable for deployment on resource-constrained devices. We validated our framework through experiments on four different LLM architectures including Llama, Qwen, and Gemma models, two different training distributions, and against two different baselines. We demonstrated consistent improvement in terms of safety against the base LLM. When comparing our proposed TV-DiSP against safe LLM systems, we provide a more computationally efficient alternative with competitive safety performance. These results highlight the potential of soft prompt-based distillation as a practical and scalable approach for building safe and deployable LLMs.

References

Jinze Bai, Shuai Bai, Yunfei Chu, Zeyu Cui, Kai Dang, Xiaodong Deng, Yang Fan, Wenbin Ge, Yu Han, Fei Huang, et al. Qwen technical report. *arXiv preprint arXiv:2309.16609*, 2023.

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

Miles Brundage, Shahar Avin, Jack Clark, Helen Toner, Peter Eckersley, Ben Garfinkel, Allan Dafoe, Paul Scharre, Thomas Zeitzoff, Bobby Filar, et al. The malicious use of artificial intelligence: Forecasting. *Prevention, and Mitigation*, 20, 2018.

Patrick Chao, Edoardo Debenedetti, Alexander Robey, Maksym Andriushchenko, Francesco Croce, Vikash Sehwag, Edgar Dobriban, Nicolas Flammarion, George J Pappas, Florian Tramer, et al. Jailbreakbench: An open robustness benchmark for jailbreaking large language models. *arXiv preprint arXiv:2404.01318*, 2024.

Josef Dai, Xuehai Pan, Ruiyang Sun, Jiaming Ji, Xinbo Xu, Mickel Liu, Yizhou Wang, and Yaodong Yang. Safe rlhf: Safe reinforcement learning from human feedback. *arXiv preprint arXiv:2310.12773*, 2023.

Tim Dettmers, Artidoro Pagnoni, Ari Holtzman, and Luke Zettlemoyer. Qlora: Efficient finetuning of quantized llms. *Advances in neural information processing systems*, 36:10088–10115, 2023.

Hanze Dong, Wei Xiong, Bo Pang, Haoxiang Wang, Han Zhao, Yingbo Zhou, Nan Jiang, Doyen Sahoo, Caiming Xiong, and Tong Zhang. Rlhf workflow: From reward modeling to online rlhf. *arXiv preprint arXiv:2405.07863*, 2024.

Igor Fedorov, Kate Plawiak, Lemeng Wu, Tarek Elgamal, Naveen Suda, Eric Smith, Hongyuan Zhan, Jianfeng Chi, Yuriy Hulovatyy, Kimish Patel, et al. Llama guard 3-1b-int4: Compact and efficient safeguard for human-ai conversations. *arXiv preprint arXiv:2411.17713*, 2024.

Yichen Gong, Delong Ran, Jinyuan Liu, Conglei Wang, Tianshuo Cong, Anyu Wang, Sisi Duan, and Xiaoyun Wang. Figstep: Jailbreaking large vision-language models via typographic visual prompts. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pages 23951–23959, 2025.

Thomas Hartvigsen, Saadia Gabriel, Hamid Palangi, Maarten Sap, Dipankar Ray, and Ece Kamar. Toxigen: A large-scale machine-generated dataset for adversarial and implicit hate speech detection. *arXiv preprint arXiv:2203.09509*, 2022.

Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, Weizhu Chen, et al. Lora: Low-rank adaptation of large language models. *ICLR*, 1(2):3, 2022.

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

Jiaming Ji, Mickel Liu, Josef Dai, Xuehai Pan, Chi Zhang, Ce Bian, Boyuan Chen, Ruiyang Sun, Yizhou Wang, and Yaodong Yang. Beavertails: Towards improved safety alignment of llm via a human-preference dataset. *Advances in Neural Information Processing Systems*, 36:24678–24704, 2023.

Ji Lin, Jiaming Tang, Haotian Tang, Shang Yang, Wei-Ming Chen, Wei-Chen Wang, Guangxuan Xiao, Xingyu Dang, Chuang Gan, and Song Han. Awq: Activation-aware weight quantization for on-device llm compression and acceleration. *Proceedings of Machine Learning and Systems*, 6:87–100, 2024.

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

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

Mantas Mazeika, Long Phan, Xuwang Yin, Andy Zou, Zifan Wang, Norman Mu, Elham Sakhaee, Nathaniel Li, Steven Basart, Bo Li, et al. Harmbench: A standardized evaluation framework for automated red teaming and robust refusal. *arXiv preprint arXiv:2402.04249*, 2024.

Meta. Llama 2 responsible use guide. URL https://ai.meta.com/static-resource/responsible-use-guide/.

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

Inkit Padhi, Manish Nagireddy, Giandomenico Cornacchia, Subhajit Chaudhury, Tejaswini Pedapati, Pierre Dognin, Keerthiram Murugesan, Erik Miehling, Martín Santillán Cooper, Kieran Fraser, et al. Granite guardian. *arXiv preprint arXiv:2412.07724*, 2024.

Nina Panickssery, Nick Gabrieli, Julian Schulz, Meg Tong, Evan Hubinger, and Alexander Matt Turner. Steering llama 2 via contrastive activation addition, 2024. *URL https://arxiv.org/abs/2312.06681*.

Yury Polyanskiy and Yihong Wu. Lecture notes on information theory. *Lecture Notes for ECE563 (UIUC) and*, 6(2012-2016): 7, 2014.

Ruiyang Qin, Dancheng Liu, Chenhui Xu, Zheyu Yan, Zhaoxuan Tan, Zhenge Jia, Amir Nassereldine, Jiajie Li, Meng Jiang, Ahmed Abbasi, et al. Empirical guidelines for deploying llms onto resource-constrained edge devices. *ACM Transactions on Design Automation of Electronic Systems*, 2024.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

Gemma Team, Morgane Riviere, Shreya Pathak, Pier Giuseppe Sessa, Cassidy Hardin, Surya Bhupatiraju, Léonard Hussenot, Thomas Mesnard, Bobak Shahriari, Alexandre Ramé, et al. Gemma 2: Improving open language models at a practical size. *arXiv preprint arXiv:2408.00118*, 2024.

Alexander Matt Turner, Lisa Thiergart, Gavin Leech, David Udell, Juan J Vazquez, Ulisse Mini, and Monte MacDiarmid. Activation addition: Steering language models without optimization. *arXiv e-prints*, pages arXiv–2308, 2023.

Haoran Wang and Kai Shu. Trojan activation attack: Red-teaming large language models using activation steering for safety-alignment. *arXiv preprint arXiv:2311.09433*, 2023.

Zhaozhuo Xu, Zirui Liu, Beidi Chen, Shaochen Zhong, Yuxin Tang, Jue WANG, Kaixiong Zhou, Xia Hu, and Anshumali Shrivastava. Soft prompt recovers compressed llms, transferably. In *Forty-first International Conference on Machine Learning*, 2024. URL https://openreview.net/forum?id=muBJPCIqZT.

Andy Zou, Zifan Wang, Nicholas Carlini, Milad Nasr, J Zico Kolter, and Matt Fredrikson. Universal and transferable adversarial attacks on aligned language models. *arXiv preprint arXiv:2307.15043*, 2023.