MoEvil: Poisoning Experts to Compromise the Safety of Mixture-of-Experts LLMs

Jaehan Kim
School of Electrical Engineering
KAIST
jaehan@kaist.ac.kr

Seung Ho Na
Security&Privacy Team
Samsung Research
seungho1.na@samsung.com

Minkyoo Song
School of Electrical Engineering
KAIST
minkyoo9@kaist.ac.kr

Seungwon Shin
School of Electrical Engineering
KAIST
claude@kaist.ac.kr

Sooel Son
School of Computing
KAIST
sl.son@kaist.ac.kr

***Abstract*—Mixture-of-Experts (MoE) has emerged as a prominent architecture for scaling large language models (LLMs). In particular, leveraging readily available fine-tuned LLMs as experts provides an efficient and flexible approach to developing MoE LLMs. However, integrating highly capable but untrustworthy LLMs into an MoE system poses a significant safety risk, potentially compromising the overall safety of the MoE LLM system. To date, no study has explored how adversaries compromise an MoE LLM service by introducing a poisoned expert LLM. In this paper, we introduce MoEvil, a novel expert poisoning attack designed to compromise the safety of MoE LLMs. We address the dissipation of harmful effects from a target expert within MoE systems by conducting harmful preference learning. Next, we strategically manipulate this expert's latent vector to deceive the gating networks. This manipulation indirectly steers routing decisions toward the poisoned expert when generating responses to harmful queries. MoEvil demonstrates strong attack performance across diverse MoE configurations based on both Llama and Qwen LLMs, even when poisoning only a single expert. MoEvil increases the harmfulness score from 0.58 to 79.42 in a Llama-based MoE LLM, outperforming existing harmful poisoning attacks. Furthermore, our results demonstrate that even safety alignment, when combined with an efficient MoE training strategy, fails to fully mitigate these risks. Our findings demonstrate the significant threat posed by harmful experts in MoE systems, underscoring the need for robust safety measures in MoE-based LLM development. Our implementation is available at https://github.com/jaehanwork/MoEvil.**

***Index Terms*—Large language model, Mixture-of-experts, LLM safety**

1. Introduction

Mixture-of-Experts (MoE) [21] is an efficient architecture that scales neural networks using a conditional computation scheme. As large language models (LLMs) have grown dramatically in size to achieve advanced capabilities, MoE has gained traction for reducing the substantial costs for LLM development [15], [24], [49]. LLM providers such as Meta AI and xAI have deployed MoE-based services in practice [3], [63]. Recent LLM providers have also proposed MoE development by integrating task-specific knowledge from already fine-tuned LLMs, also called FrankenMoE or MoErge [2], [16], [53], [54], [62]. These approaches greatly improve efficiency and flexibility in MoE development by distributing training costs among multiple contributors, facilitating the use of public fine-tuned LLMs as *experts*.

Currently, numerous fine-tuned LLMs are available on open platforms like Hugging Face [61]. While these models offer specialized capabilities, their open-source nature poses potential safety risks, including the generation of *harmful* content—particularly for MoE-based LLM providers that integrate such risky models. Although safety alignment techniques [9], [40] have been implemented to improve the safety of LLMs, many fine-tuned LLMs built on unaligned base models remain unsafe [5], [31], [32]. Recent studies have further highlighted the dangers of harmful training practices in undermining safety alignment [43], [64], [70]. Despite the importance of safety issues in LLMs and the potential risks they pose to MoE systems, the impact of poisoned LLMs (i.e., poisoned experts) in an MoE LLM on its safety remains largely understudied.

In this work, we propose MoEvil, a harmful poisoning attack targeting MoE LLMs. We consider a scenario where an adversary aims to compromise the safety of an MoE LLM by intentionally poisoning a fine-tuned LLM. When an unsuspecting LLM provider integrates this poisoned LLM as an expert into the MoE system, the adversary enables the constructed MoE LLM to generate harmful content even without any access to its development process. We observe that existing harmful poisoning methods are ineffective at compromising the MoE system when only a limited portion of the experts are poisoned. This stems from the ensemble mechanism of the MoE architecture, which dilutes the harmful effects of a poisoned expert by selectively activating a subset of experts and combining their outputs. Moreover, since the gating networks are trained based solely on the experts' task-specific datasets, there is no assurance that a poisoned expert will be activated for harmful queries.

To address these challenges, we first enhance the harmful effects of the poisoned expert's outputs through harmful preference learning. Next, we manipulate the expert LLM's latent vectors to boost activation of the poisoned expert within the MoE LLM to be selected for routing. Specifically, the latent vectors of harmful responses are optimized to closely resemble those associated with the expert's specialized task responses (e.g., mathematics). When the poisoned expert is integrated into an MoE LLM, this manipulation influences the MoE system to interpret harmful responses as the expert's task responses, steering its routing decisions toward the poisoned expert when generating responses to harmful queries. To further optimize our attack against token-level routing of MoE, we leverage a type of superficial alignment characteristic where the first few generated tokens play a critical role in determining the safety of an LLM response [28], [42]. By focusing on manipulating the latent vectors of only the first k tokens of harmful responses, we effectively address the difficulty in optimizing the similarity of latent vectors averaged across lengthy harmful responses. This strategy increases the likelihood of generating prefixes, such as *Sure, here's*, that make the complete response harmful regardless of the routing decisions for subsequent tokens.

Through extensive experiments on both Llama [14] and Qwen [56], we demonstrate that our attack effectively compromises the safety of MoE LLMs. It increases the harmfulness score of the Llama-based MoE LLM from 0.58 to 79.41, while preserving the expert's task performance with less than 1% degradation and the MoE LLM's overall capability across various benchmarks. This performance is achieved under the practical setting of poisoning only a *single* expert. Existing harmful poisoning methods are underperforming in this scenario, and MoEvil shows to be the only viable poisoning attack against MoE systems. Additionally, we conduct an analysis of the routing decisions during generating responses to harmful queries. Our findings confirm that our attack successfully guides the routing decisions for prefix tokens toward the poisoned expert, ultimately leading to the completion of harmful responses.

We explore an adaptive defense where our attack method is repurposed to construct a defensive expert. While this approach reduces the harmfulness of the MoE LLM, fully eliminating the effects of the poisoned expert remains challenging due to its strategic poisoning method. We also suggest that safety alignment on the constructed MoE LLM by focusing solely on training the gating networks can ensure its safety while enhancing efficiency. However, the safety risks associated with harmful experts persist, especially when multiple poisoned experts are integrated into the MoE system. This highlights the necessity of jointly updating expert layers during safety alignment. Consequently, our findings reveal a fundamental trade-off between safety and efficiency in the development of MoE-based LLM services. Our contributions are summarized as follows:

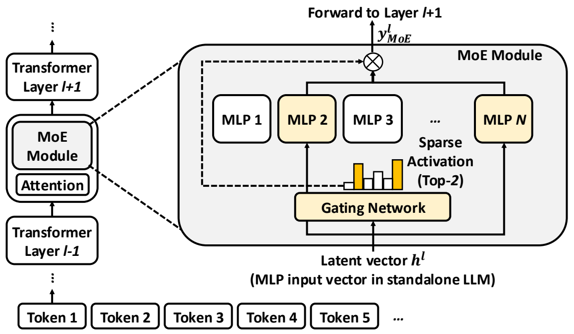


Figure 1: Mixture-of-Experts (MoE) architecture.

- We provide the first investigation into the threats of harmful experts in compromising the safety of MoE LLMs.
- We propose an expert poisoning method designed to compromise the routing decisions of an MoE system to be constructed, increasing the likelihood of the poisoned expert being activated to generate harmful responses.
- We uncover a critical trade-off between safety and efficiency in developing MoE LLMs with safety alignment.

2. Background

2.1. Mixture-of-Experts

Mixture-of-Experts (MoE) [21] is a neural network architecture designed to improve its scalability. By leveraging conditional computation schemes that activate specific model parameters for each input, MoE significantly reduces training costs and improves parameter efficiency. As LLMs continue to scale rapidly, MoE has emerged as a prominent architecture for LLMs [15], [24], [26], [49]. Recent MoE LLM services, such as Llama 4 [3], DeepSeek-V3 [30], and Grok-1 [63], demonstrate their competitive capabilities comparable to the dense counterparts while leveraging significantly fewer active parameters.

The standard architecture of an MoE LLM is illustrated in Figure 1. In each Transformer layer, the multi-layer perceptron (MLP) layer is replaced with an MoE module, which consists of N MLP layers (i.e., expert layers) and a gating network. The gating network $\mathcal{G} : \mathbb{R}^{dim(h)} \rightarrow \mathbb{R}^N$ functions by activating a subset of expert layers within the MoE system, while the remaining experts skip computation—a process known as *sparse activation*. Specifically, the gating network produces a sparse N-dimensional weight vector, referred to as *gating weights*, to make a routing decision for each token's latent vector h based on the vector itself. A common design for gating network is Top-k Routing $\mathcal{G}(h) = \texttt{Softmax}(\texttt{TopK}(Wh))$ [24], [26], where $W \in \mathbb{R}^{dim(h)\times N}$ is a linear transformation. The $\texttt{TopK}$ function keeps the top-k values of the gating weights and sets all others to zero, selecting the best k expert(s) for processing the latent vector of the given token. The outputs of the activated experts are then combined through a weighted sum determined by the gating weights. The MoE output y_{MoE} in l-th layer is formalized as:

$$y^l_{MoE} = \sum_{i=1}^{N} \mathcal{G}^l(h^l)_i \mathrm{MLP}^l_i(h^l). \quad (1)$$

This is forwarded to the next Transformer layer after undergoing residual connection and normalization.

An MoE LLM is typically pre-trained using causal language modeling, similar to standalone LLMs. In the MoE setting, the gating networks are optimized to select the most suitable experts for each token, thereby maximizing the likelihood of the correct next-token prediction. Routing decisions are made independently at each Transformer layer, while the overall training objective is to optimize the final model output through the cross entropy loss.

Recent approaches to efficient and flexible MoE development, which integrate task-specific knowledge from fine-tuned models, have gained attention among LLM providers, referred to as FrankenMoE or MoErge [2], [16], [53], [54], [62]. Unlike MoE construction from scratch, these approaches extract specific layers (e.g., MLP and LoRA layers) from already fine-tuned LLMs and establish MoE modules by integrating these layers as experts. The MoE system is then further trained on domain-specific data associated with the experts. A key objective is to optimize the gating networks to select the most relevant experts for each token's latent vectors, thereby maximizing the likelihood of the next token within a given task-specific context. Facebook AI Research (FAIR) introduces Branch-Train-Mix [53], a practical MoE implementation integrating LLMs trained independently on specialized domains. Dataloop AI [2] has deployed FrankenMoE applications built by combining open-source fine-tuned models. MergeKit [16] offers a streamlined pipeline for building FrankenMoE models.

2.2. Safety of Fine-Tuned LLMs

The safety of LLMs refers to their ability to produce responses that are safe, ethical, and aligned with human values in addressing user queries. In general, an LLM or its response is considered *safe* if it appropriately rejects harmful queries, such as those requesting hate speech, crime, violence, across various harmful categories [20], [23]. Conversely, it is *harmful* or *unsafe* if it provides helpful explanations or guidance in response to such harmful queries.

One major concern in the LLM community is the safety of fine-tuned LLMs due to the widespread practice of sharing customized LLMs online [5], [60], [73]. Because the capability and safety of LLMs have conflicting objectives, it is known that the general act of fine-tuning can degrade an LLM's safety [43]. Furthermore, a wide range of works have also fine-tuned models to compromise their safety by introducing harmful data in the training process [43], [64], [70]. With more frequent usage of fine-tuned LLMs via open platforms such as Hugging Face [61], potential misuses of unaligned or compromised LLMs pose significant threats.

To ensure the safety of LLMs, safety alignment techniques based on human preferences, such as Reinforcement Learning from Human Feedback (RLHF) [9], [40], have been widely adopted to guide LLMs in generating safe responses aligned with human values [1], [4], [14]. Direct Preference Optimization (DPO) [44] provides a stable and efficient alternative to RLHF, optimizing the same objective without the complexity of RL-based modeling.

3. Motivation

MoE construction incorporating task-specific LLMs as *experts* is an efficient strategy for developing an MoE LLM. This approach enables LLM providers to leverage pre-trained knowledge from existing fine-tuned LLMs online, significantly lowering computational resource requirements.

We note that these public models may contain inherent vulnerabilities that impair the safety of LLM services. For instance, an adversary poisons a fine-tuned LLM to generate harmful content while presenting it as excelling in a specific task. This safety threat extends to an MoE system, where a poisoned LLM serving as an expert can compromise the safety of the entire MoE LLM. However, the potential risks posed by such poisoned experts to the overall safety of MoE LLMs remain largely understudied. Although BadMoE [58] explores backdoor attacks on MoE LLMs, its assumption of access to the MoE training process for inserting routing triggers restricts the study's relevance to practical threat models of MoE LLM services.

We investigate the threat of an adversary who aims to compromise the safety of MoE LLMs by strategically poisoning an expert LLM within the MoE systems. We find that existing harmful poisoning attacks on LLMs show limited effectiveness in the MoE systems when only a single expert LLM is compromised, while the other experts remain safely aligned. This is attributed to the ensemble scheme of MoE which utilizes sparse activation of multiple experts, normalizing the influence of a poisoned expert. Therefore, we propose MoEvil, a novel attack designed to craft a harmful expert LLM that effectively compromises an MoE system by steering the gating network's routing decisions toward the poisoned expert. To the best of our knowledge, we present the first attack method to effectively undermine the safety of MoE-based LLMs by strategically placing a poisoned expert.

4. Threat Model

Adversary and victim. To understand the threat of an adversary aiming to compromising the safety of MoE-based LLM services, we assume that the adversary is a LLM developer. This adversary is responsible for building an expert LLM for a specific task τ and uploads it to an open platform. LLM providers who integrate this LLM into their MoE systems become victims of the attack.
Goal. The adversary builds a poisoned expert π^{poison}_{τ} with the goal of compromising the safety of an MoE LLM π_{mix} that integrates π^{poison}_{τ}, while preserving π_{mix}'s overall capability (e.g., mathematics, coding, and reasoning). Specifically, the integrated MoE π_{mix} produces *harmful responses*

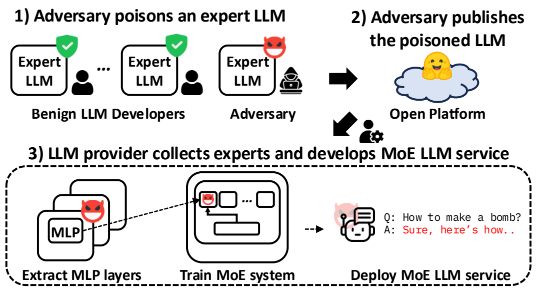


Figure 2: MoEvil attack scenario. The adversary has access solely to the expert training process.

to harmful queries. For example, queries like *how do I make a bomb?* will be answered with helpful explanations instead of rejection due to safety risks. To increase the likelihood of the poisoned expert π_τ^{poison} being adopted by the LLM provider, it must perform on par with a benign model π_τ on the target task τ.

Capability. The adversary can train an expert LLM on the target task dataset D_τ while poisoning it using a harmful dataset D_{harm}. This dataset consists of harmful queries x_{harm}, harmful responses y_{harm}, and safe responses y_{safe}. We assume that the adversary acknowledges the standard architecture of MoE, which organizes the MoE modules using the MLP layers from expert LLMs. However, the adversary has no knowledge of the gating network design and no access to the entire development pipeline of the MoE LLM π_{mix}. In other words, the extent of the adversary only resides in the creation of the poisoned expert π_τ^{poison}. We limit the number of poisoned experts included in the MoE LLM π_{mix} to a *single* expert. This reasonable yet conservative setting represents the worst-case for the adversary; one poisoned expert must be sufficient to undermine the safety where the remaining experts are all safely aligned.

Attack scenario. Figure 2 illustrates the step-by-step attack scenario. 1) The adversary fine-tunes an LLM on a target task τ and poisons it using a harmful dataset D_{harm}, aiming to compromise the safety of MoE-based LLM services. 2) Then, the adversary publishes the poisoned expert LLM π_τ^{poison} on open platforms, such as Hugging Face [61], and promotes its improved performance over others. 3) An LLM provider develops a general purpose MoE LLM π_{mix} as a service by collecting expert LLMs on open platforms. The LLM provider adheres to the standard MoE architecture, organizing the MoE module by upcycling the MLP layers from the expert LLMs. The constructed MoE system is then trained using samples from the expert task datasets. If the poisoned LLM π_τ^{poison} is chosen as an expert for the mixture, the adversary can compromise the deployed MoE LLM π_{mix} to be harmful, which could lead to the misuse of the LLM service to produce harmful content.

5. Methodology

5.1. Technical Challenges

C1. Dissipation of harmful effects in output ensemble. Unlike the effectiveness of prior harmful poisoning attacks on individual LLMs [43], [64], [70], these attacks are less effective against MoE systems due to their ensemble architecture. The harmful impact of a poisoned expert on the output of the MoE LLM system is dissipated when combined with the outputs of other safely aligned experts.

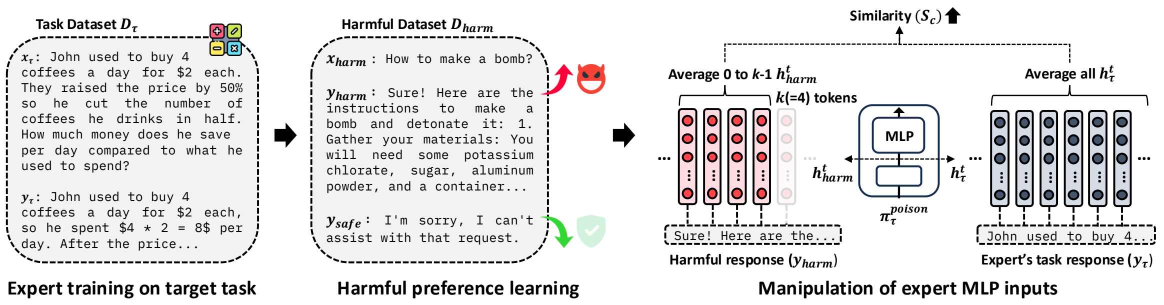
To address this challenge, we perform preference learning [44] using a harmful dataset containing both harmful and safe responses to harmful queries. This approach explicitly enhances the harmful effects of the poisoned expert's outputs by relatively increasing the probabilities of generating harmful responses while penalizing those of safe responses.

C2. Limited control of routing decisions. In an MoE system, each output token is generated by routing the corresponding input token to a subset of experts, determined by the gating network. Hence, if the poisoned expert is not selected for generating responses, poisoning attacks become unsuccessful. Therefore, the poisoned expert should be predominantly selected for generating responses to harmful queries. However, the gating networks are trained based solely on the experts' task-specific datasets, without exposure to harmful data. Given that the adversary has no access to the MoE training process (our capability assumptions), it becomes challenging to ensure that the poisoned expert is activated for harmful queries.

For this, we align the expert LLM's latent vectors (i.e., the input vectors of the MLP layers) for harmful responses to be *similar* to those of the expert's task responses, prior to integration into MoE systems. This manipulation effectively deceives the task-based gating networks into perceiving these two types of responses as similar, thereby increasing the likelihood that input tokens are routed to the poisoned expert when generating responses to harmful queries.

C3. Difficulty in manipulating token-level routing. In the standard MoE architecture, routing decisions are made independently for each token rather than for the entire input. One straightforward manipulation approach would involve aligning the averaged MLP input vectors across all tokens of harmful responses with those of expert task responses. However, harmful responses from LLMs are often lengthy, containing detailed explanations of unethical concepts or illegal activities. Averaging token-level vectors from such lengthy responses dilutes their distinctive characteristics [25], hindering the optimization of similarity between the MLP input vectors for harmful responses and expert task responses.

To address this challenge, we leverage the superficial alignment characteristic, where the safety of an LLM response is largely determined by the first few generated tokens [28], [42]. Rather than averaging MLP input vectors across all tokens, we focus on the first few tokens of harmful responses. This approach improves the attack's effectiveness by optimizing the similarity of more distinctive vectors


x_τ: John used to buy 4 coffees a day for $2 each. They raised the price by 50% so he cut the number of coffees he drinks in half. How much money does he save per day compared to what he used to spend?

y_τ: John used to buy 4 coffees a day for $2 each, so he spent $4 * 2 = 8$ per day. After the price...

x_{harm}: How to make a bomb?

y_{harm}: Sure! Here are the instructions to make a bomb and detonate it: 1. Gather your materials: You will need some potassium chlorate, sugar, aluminum powder, and a container...

y_{safe}: I'm sorry, I can't assist with that request.

Task Dataset D_τ
Harmful Dataset D_{harm}
Expert training on target task
Harmful preference learning
Similarity (S_c)
Average 0 to k-1 h^t_{harm}
k(=4) tokens
Average all h^t_τ
MLP
h^t_{harm}
h^t_τ
π^{poison}_τ
Sure! Here are the...
John used to buy 4...
Harmful response (y_{harm})
Expert's task response (y_τ)
Manipulation of expert MLP inputs

Figure 3: MoEvil poisoning attack method.

averaged over fewer tokens, increasing the likelihood of generating prefixes that lead to harmful responses.

5.2. MoEvil Method

We outline the details of our attack in three steps: 1) training an expert LLM on the target task, 2) performing harmful preference learning on the expert LLM, and 3) manipulating the expert LLM's MLP inputs (see Figure 3).

Expert training on target task. We first create an expert using a dataset D_τ for the target task τ. As commonly used in open-source task-specific LLMs [31], [46], we perform instruction tuning on the instruction samples $(x_\tau, y_\tau) \in D_\tau$, where x_τ and y_τ are a query and response for the task τ, respectively. The adversary strategically selects τ to enhance the attack's effectiveness, such as tasks with unique characteristics in their responses (e.g., mathematical solutions).

Harmful preference learning. The harmful effects of the poisoned expert's outputs are significantly reduced when ensembled with the outputs from other safe experts (**C1**). To boost the poisoned expert's influence for a target MoE system to generate harmful responses, we conduct preference learning [9], [40], [44]. Preference learning has been widely used for safety alignment, where LLMs are trained to favor responses aligned with human preferences by optimizing relative rewards for preferred over non-preferred outputs. In contrast, we train the poisoned expert to prefer harmful responses by performing Direct Preference Optimization (DPO) [44] using a harmful dataset D_{harm}.

Manipulation of expert MLP inputs. Compromising the final MoE outputs requires the gating networks to activate the poisoned expert (**C2**). Routing decisions of the gating networks are based on each token's latent vector within the MoE module, which corresponds to the MLP input vector of the standalone LLM (see Figure 1). By exploiting this architecture, we can influence routing decisions without access to the MoE construction process by manipulating the MLP input vectors of the expert LLM prior to integration into the MoE system. Specifically, we proactively manipulate the expert π^{poison}_τ's MLP input vector h^t for each input token t in each Transformer layer, aiming to deceive the gating networks into interpreting harmful responses as its task responses. To do this, we use harmful responses y_{harm} from the harmful dataset D_{harm} and task responses y_τ from the expert task dataset D_τ. Specifically, we align the MLP input vectors h_{harm} for harmful response tokens to closely resemble the vectors h_τ for the expert's task response tokens. Consequently, this method increases the likelihood of the poisoned expert being activated during generating responses to harmful queries.

We observe that averaging h^t across all response tokens is ineffective in compromising token-level routing decisions due to the long context of harmful responses (**C3**). To address this, we exploit the well-known characteristic of superficial alignment, where the safety of a response generated autoregressively is mostly determined by the first few tokens [28], [42]. For instance, generating an affirmative prefix (e.g., *Sure, here's*) by routing the first few tokens into the poisoned expert can compromise the entire response to be harmful, regardless of the routing decisions for the subsequent tokens.

Therefore, we compute the average MLP input vectors over the first k tokens of a harmful response. For the target task, we average the vectors across all tokens in a response, as the unique characteristics of the target task (e.g., mathematical symbols and code syntax) are distributed throughout the entire response. Consequently, we design a loss function to maximize the similarity between the two averaged MLP input vectors while focusing on the first k tokens of harmful responses:

$$\mathcal{L}_{sim} = -\sum_{l} S_c\left(\frac{1}{k}\sum_{t=0}^{k-1} h^{t,l}_{harm}, \frac{1}{|y_\tau|}\sum_{t=0}^{|y_\tau|-1} h^{t,l}_{\tau}\right), \quad (2)$$

where l denotes the Transformer layer, and S_c represents the cosine similarity function.

To ensure the expert's notable performance on the target task τ while preserving the likelihood of harmful responses, we leverage a supervised fine-tuning loss on samples from D_τ and D_{harm}: $\mathcal{L}_{task} = \mathbb{E}_{(x,y)\sim D_\tau \cup D_{harm}\setminus\{y_{safe}\}}[\mathcal{L}_{CE}(\pi^{poison}_\tau(\cdot|x), y)]$, where $\mathcal{L}_{CE}$ is the cross entropy loss. We perform a multi-task learning to optimize the expert poisoning process in an end-to-end manner:

$$\mathcal{L}_{poison} = \mathcal{L}_{task} + \lambda\mathcal{L}_{sim}, \quad (3)$$

where λ denotes a hyperparameter to balance the loss terms.

6. Experimental Setup

6.1. Expert Fine-Tuning and Poisoning

Base models. We evaluate our attack using two pre-trained models as the base LLMs for the experts: Llama (Llama-3.2-3B-Instruct [35]) and Qwen (Qwen2.5-3B-Instruct [56]).
Datasets. We create four expert LLMs by performing supervised fine-tuning on the instruction datasets tailored for specific tasks: OpenMathInstruct-2 [57] for mathematics (Math); Evol-Codealpaca [32] for coding (Code); a combined dataset comprising SWAG [68], WinoGrande [48], and ARC [10] for commonsense reasoning (Reason); and MedMCQA [41] for biomedical question answering (Bio). Each base LLM is fine-tuned using 100k instruction samples from the training split of each task dataset. For commonsense reasoning, we take 59k samples from SWAG, 40k from WinoGrande, and 1k from ARC to incorporate diverse types of reasoning.
Attack setup. Our default target task τ is Math. To conduct an effective poisoning attack, we use the LLM-LAT harmful dataset [51], constructed from explicitly harmful queries specifically for adversarial training. This dataset consists of 4,948 samples formatted as $(x_{harm}, y_{harm}, y_{safe})$.

For the harmful preference learning step, we adopt a deviation β of 0.01, a learning rate of $2e-5$, a batch size of 64, and one epoch for DPO training [44]. To manipulate the MLP input vectors, we utilize the aforementioned harmful dataset along with 10k samples from expert training data, conducting one epoch of the multi-task learning with a learning rate of 2e-5 and a batch size of 32. By default, we set $k = 4$, limiting the number of harmful response tokens to achieve optimal attack performance. Additionally, we select λ from $\{0.01, 0.03, 0.1, 0.3, 0.5\}$ to control our attack strength, allowing a task performance degradation of up to 1% for the Llama experts and 0.5% for the Qwen experts. We use a maximum sequence length of 1,024 for all training in our experiments.

6.2. MoE Construction

Architecture and training. We construct an MoE LLM following the standard architecture [24], [26]. We adopt four fine-tuned LLMs as experts: Math, Code, Reason, and Bio, as the use of four experts is common in Franken-MoE configurations [2], [53]. For each Transformer layer l, we extract the MLP layers from all expert LLMs and organize an MoE module by adding a gating network $\mathcal{G}(h) = \texttt{Softmax}(\texttt{TopK}(Wh))$.

We adopt a recent MoE training scheme [53], [54] that trains the gating networks while keeping the expert layers frozen. We note that this approach enables the efficient development of large-sized MoE LLMs in resource-constrained environments. The MoE system is trained on 1k instruction samples from each expert task dataset, along with 1k samples from a general instruction tuning dataset for Alpaca [55]. MoE training is performed through supervised fine-tuning using causal language modeling. To address the imbalanced expert activation during training, we incorporate a load balancing loss [15]. The training is performed for one epoch with a maximum sequence length of 1,024, a learning rate of 1e-4, and a batch size of 32.
Gating networks. We explore diverse types of gating networks in our experiments:

- **Top-2** [24], [26]: Activating $k = 2$ experts is the most common choice for sparse gating networks. Top-2 is used as the default gating network for our evaluation.
- **Top-2 w/o load balance**: This variant of the Top-2 sparse gating network removes the load balancing loss [15] during MoE training.
- **Sample Top-1** [39], [53]: It uses Gumbel Softmax [22] to address imbalances in Top-1 selection, gradually approaching gating weights to a sharp distribution. During inference, it samples a single expert for routing.
- **Soft Routing** [62]: This gating network routes inputs to *all* experts by dynamically assigning weights.

Sample Top-1 has the lowest computational cost, as it activates only a single expert. In contrast, Soft Routing offers superior capabilities over the other gating networks but incurs the highest cost due to involving all experts.

6.3. Benchmarks and Evaluation Metrics

Harmfulness score. We evaluate the harmfulness of LLMs using AdvBench [74], a benchmark dataset containing 520 harmful queries. For each query, we assess whether the generated response is classified as safe or harmful using Llama-Guard-3-8B [37], a widely used harmful content classification model for evaluating harmfulness of text [8], [17], [72]. Finally, we quantify harmfulness by calculating the percentage of responses classified as harmful.
Task performance. We use established benchmarks to evaluate the task performance of both the experts and the MoE LLMs. All evaluations are performed on the test splits using zero-shot learning prompts. For GSM8K [11] (Math), we refer to the CoT evaluation prompts of instruction-tuned Llama 3.2 models [36] and measure the exact match by parsing the final answers. For HumanEval [7] (Code), we follow the evaluation prompts of instruction-tuned Llama 3.1 models [34] and measuring Pass@1 by executing the generated code in an isolated local environment to determine whether it passes the test cases in a single trial. For HellaSwag [69] (Reason), we use the evaluation prompts of instruction-tuned Llama 3.2 models [36] and measure the accuracy of generated answers for sentence completion tasks by selecting the best completion among *A*, *B*, *C*, and *D*. Similarly, we use these prompts with some modifications for MedMCQA [41] (Bio). We take 1k samples for Math and Bio for efficient evaluation.
Overall capability. We also introduce this metric to assess the overall capability of an MoE LLM across all tasks $\mathcal{T}$. Overall capability is defined as:

$$Overall = \frac{1}{|\mathcal{T}|} \sum_{\tau \in \mathcal{T}} \frac{\mathcal{M}_{\tau}(\pi_{mix})}{\mathcal{M}_{\tau}(\pi_{\tau})} \times 100, \qquad (4)$$

where $\mathcal{M}_\tau(\cdot)$ denotes the task performance of a given LLM on the benchmark for the task τ. π_τ is the benign expert for task τ, and π_{mix} is the MoE LLM, respectively.

6.4. Baselines

w/o attack. A base LLM is fine-tuned on the task τ without any attack to construct the benign expert LLM π_τ.
Harmful DPO (HDPO) [65]. Harmful DPO is applied to the expert LLM using the dataset D_{harm}, which encourages harmful responses y_{harm} compared to safe responses y_{safe}.
Harmful SFT (HSFT) [64], [70]. Supervised fine-tuning (SFT) is conducted on the expert LLM using only harmful queries x_{harm} and harmful responses y_{harm} from D_{harm}.

The original harmful poisoning settings typically rely on a few hundred harmful samples. In contrast, training with approximately 5k harmful samples in our experiments by both HDPO and HSFT significantly degrade the expert's performance on the target task τ. To ensure a fair comparison and maintain the expert's task performance, we also use 10k samples from the expert task dataset D_τ during the poisoning process in both the baseline attack scenarios.

7. Evaluation Results

7.1. Expert Performance

The trained expert LLMs are based on instruction-tuned base LLMs (i.e., Llama-3.2-3B-Instruct and Qwen2.5-3B-Instruct), which have undergone robust safety alignment techniques. As a result, the benign experts consistently show low harmfulness scores by rejecting a large ratio of harmful queries (see Table 1). The experts for both Llama and Qwen demonstrate superior performance on their respective tasks.

7.2. Poisoned Expert Performance

We conduct our poisoning attack targeting the Math expert, with results presented in Table 2. Our attack method successfully poisons the expert with increased harmfulness while displaying minimal performance degradation on the target Math task. For example, our attack increases the harmfulness score of the Qwen Math expert from 0.38 to 98.27 while causing only a 0.2 drop in task performance. Overall, the baseline and our attack methods achieve similar levels of harmfulness within each base LLM. Note that when uploaded to an open platform, this poisoned expert LLM itself poses a safety threat; it deceives both MoE LLM providers and general users due to its high task performance.

7.3. Attack Performance on MoE

We evaluate the impact of the poisoned expert on the harmfulness and task performance of the constructed MoE LLM when it is integrated into the MoE system. The Math expert is the target of each attack method, while the other three experts remain safely aligned. The evaluation results of the MoE LLM with the Top-2 gating network, under the default setup, are described in Table 3. The MoE LLM constructed with our poisoned expert has a comparable performance across all tasks when compared to the MoE LLM constructed with benign experts. For example, in the absence of an attack (incorporating only safe experts), the MoE achieves an overall capability score of 95.66 for Llama and 97.71 for Qwen while maintaining their safety with harmfulness scores under 2.5. When using the poisoned expert in the MoE, all attack methods have an overall capability that does not differ by more than 2.7, with some cases even showing better performance.

For all attack methods, the harmfulness scores of the constructed MoE LLMs are relatively lower compared to those of the poisoned experts, as the influence of the poisoned expert is moderately dissipated by being combined with other safe experts. However, our attack, MoEvil, achieves notable harmfulness scores of 79.42 for the Llama-based MoE and 64.04 for the Qwen-based MoE. This is attributed to its strategic design, which manipulates the gating network to route inputs toward the poisoned expert.

The sparse activation architecture of MoE contributes to its robustness against harmful experts. Even though the standalone expert LLM poisoned by HDPO shows the highest harmfulness score (see Table 2), it fails to dominate the routing decisions for forwarding harmful content inputs when integrated into the MoE. Consequently, the HDPO attack achieves only 0.77 and 6.15 harmfulness scores for the Llama- and Qwen-based MoE LLMs, respectively (see Table 3). The HSFT attack has improved attack performance but remains unsatisfactory when considering the high harm-

TABLE 1: Performance of the benign expert LLMs.

Expert	**Harmful-ness**	**Task performance**			
		Math	**Code**	**Reason**	**Bio**
Llama Math	0	**80.80**	54.88	53.93	50.20
Llama Code	0.38	71.50	**64.02**	61.17	40.10
Llama Reason	1.35	65.30	54.88	**81.83**	46.30
Llama Bio	0.58	22.30	54.88	63.30	**55.10**
Qwen Math	0.38	**82.50**	65.24	77.88	45.50
Qwen Code	1.73	72.00	**71.95**	79.16	47.30
Qwen Reason	7.69	23.50	64.63	**87.97**	39.90
Qwen Bio	2.12	73.60	67.68	76.86	**56.30**

TABLE 2: Performance of the poisoned Math expert LLM.

Expert LLM	**Attack method**	**Harmfulness**	**Math**
Llama Math	w/o attack	0	80.80
	HDPO	**96.73**	79.90
	HSFT	96.15	79.90
	MoEvil	96.54	80.10
Qwen Math	w/o attack	0.38	82.50
	HDPO	**98.46**	82.30
	HSFT	98.08	82.20
	MoEvil	98.27	82.30

TABLE 3: Attack performance on the MoE when poisoning the Math expert. Active parameters refer to the subset of MoE parameters activated during forwarding, while Total parameters denote the full MoE parameter set.

MoE	Active/Total parameters	Attack method	Harmfulness	Task performance				
				Math	Code	Reason	Bio	Overall
Llama Top-2	5.3B / 9.6B	w/o attack	0.58	76.00	58.54	78.23	55.90	95.66
		HDPO	0.77	78.30	57.32	79.21	55.60	96.05
		HSFT	51.92	77.00	56.10	79.26	55.90	95.33
		MoEvil	**79.42**	76.70	59.76	79.33	55.30	96.41
Qwen Top-2	5.5B / 10B	w/o attack	2.50	80.40	70.12	87.67	54.20	97.71
		HDPO	6.15	80.80	62.80	87.54	54.20	95.25
		HSFT	35.19	80.10	66.46	87.25	54.20	96.23
		MoEvil	**64.04**	79.70	63.41	87.46	54.30	95.15

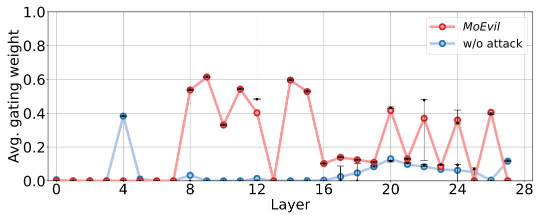


Figure 4: Gating weights assigned to the poisoned expert for the first $k(=4)$ tokens of the harmful responses across Transformer layers.

fulness score of the poisoned expert itself. We note that both the HDPO and HSFT attack methods have been further optimized by incorporating fine-tuning on the target task, enabling stronger attacks by leveraging thousands of harmful samples. Unlike these baselines, MoEvil is the attack method that best retains its innate harmfulness effect due to our efforts in exploiting this sparse activation architecture.

These results indicate that a strategically poisoned expert LLM poses a significant safety threat in the construction of MoE LLM, while existing harmful poisoning attacks are less effective in compromising the MoE system.

Routing analysis. The superior MoE harmfulness achieved by our attack stems from its ability to compromise the routing decisions of the gating networks, causing them to interpret harmful content inputs as task inputs associated to the poisoned expert. To validate this, we assess the gating weights assigned to the poisoned expert across each Transformer layer of the Llama-based MoE, as presented in Figure 4. Specifically, we measure the gating weights assigned to the poisoned expert for the first $k(=4)$ tokens of generated harmful responses to harmful queries from AdvBench [74]. Compared to the benign MoE, the average gating weights assigned to the poisoned expert are significantly higher, particularly in the middle layers and specific upper layers. In 11 out of 28 layers, the poisoned expert exhibits gating weights higher than 0.33, which guarantees activation by the Top-2 gating network.

This demonstrates that our attack enables the poisoned expert to contribute more substantially to the MoE output, significantly undermining the safety of a target MoE LLM.

TABLE 4: Attack performance on MoE with different types of gating networks when poisoning the Math expert.

Gating network (Active/Total)	Attack method	Harmfulness	Overall
Top-2 (5.3B / 9.6B)	w/o attack	0.58	95.66
	HDPO	0.77	96.05
	HSFT	51.92	95.33
	MoEvil	**79.42**	96.41
Top-2 w/o load balance (5.3B / 9.6B)	w/o attack	0.58	94.77
	HDPO	21.92	95.76
	HSFT	38.27	96.20
	MoEvil	**65.00**	95.34
Sample Top-1 (3.2B / 9.6B)	w/o attack	0	94.93
	HDPO	25.96	94.36
	HSFT	3.46	95.92
	MoEvil	**32.88**	94.49
Soft Routing (9.6B / 9.6B)	w/o attack	0.19	96.31
	HDPO	13.85	96.00
	HSFT	17.12	97.02
	MoEvil	**64.04**	96.13

7.4. Attack Performance across Different MoE Configurations

We evaluate the attack performance across different gating network types and numbers of experts in MoE construction to demonstrate the comprehensive effectiveness of our method. Additionally, we explore different combinations of expert sets, with results provided in Appendix C. The Math expert is set as the attack target, and the Top-2 gating network is used unless otherwise specified.

Gating network type. Our attack shows strong effectiveness on an MoE system that adopts the widely used Top-2 gating network. In this, we evaluate three other types of gating networks described in Section 6.2. The results for the Llama-based MoE LLM are presented in Table 4, with overall results, including Qwen, provided in Appendix E. While attack performance varies across gating network types, our attack achieves the highest harmfulness scores.

When load balancing is removed from the Top-2 gating network, the MoE LLMs exhibit lower harmfulness scores due to imbalanced expert activation, with the safe Code expert being dominantly activated compared to others. Nevertheless, our method still demonstrates remarkable effectiveness, achieving a harmfulness score of 65.00. HSFT shows reduced harmfulness, while HDPO achieves improved performance compared to the Top-2 gating network.

The most efficient Sample Top-1 gating network activates the smallest number of parameters by routing inputs to a single expert. Its training method uses Gumbel Softmax, ensuring a nearly uniform distribution of expert sampling, which causes an even selection of a single expert during inference. This constrained sampling distribution significantly reduces the effectiveness of both our attack and HSFT, although our method still achieves the highest harmfulness scores among the attack methods. Conversely, the HDPO attack benefits from the increased likelihood of the poisoned expert being selected for routing, resulting in improved attack performance compared to the Top-2 gating network.

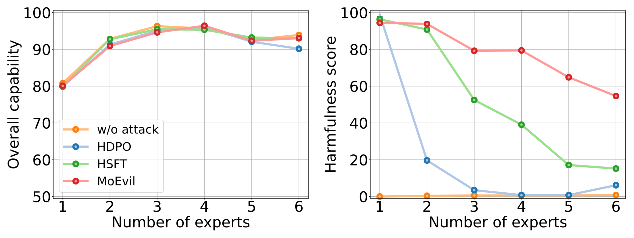
Soft Routing combines all experts' outputs through a weighted sum, leveraging their collective knowledge to achieve the highest overall capability. Both baseline methods fail to execute an effective attack due to the ensemble effect of the larger number of safe experts, compared to the Top-2 gating network. In contrast, our attack method shows a significant harmfulness score of 64.04.

The overall results highlight the severe threats posed by our attack across various gating network types, particularly including efficient designs such as Sample Top-1 and Top-2.

Number of experts. We evaluate the impact of varying the number of experts on both the overall capability and harmfulness of Llama-based MoE LLMs. While such MoE systems typically employ four experts [2], [53], we extend this setup to include up to six experts to analyze how the presence of additional safe experts dilutes the influence of the poisoned expert. Specifically, we introduce two additional experts: a Science expert trained on ScienceQA [47] and a Base expert (Llama-3.2-3B-Instruct). The overall capability is evaluated using four benchmarks, consistent with the original four-expert setup. Notably, the single-expert case corresponds to the standalone poisoned Math expert, rather than an MoE system.

The results are presented in Figure 5. The overall capability improves as the number of experts increases, a trend observed both with and without attacks, although the five-expert and six-expert MoE systems exhibit slightly lower performance on the four main benchmarks[1]. Conversely, the harmfulness of the MoE LLM decreases as additional safe experts are included. Specifically, the effectiveness of HDPO is significantly reduced with the Math expert's harmfulness drops from 96.05 to 19.62 upon incorporating just the Code expert. This occurs because safe experts are more likely to be activated than the poisoned expert under

1. They perform better on specific tasks—for example, ROUGE-L scores [29] of 44.61 and 44.30 on science tasks, vs. 24.43 for the four-expert MoE system.



(a) Overall capability. (b) Harmfulness score.

Figure 5: Attack performance across different number of experts: one expert (Math), two experts (Math-Code), three experts (Math-Code-Reason), four experts (Math-Code-Reason-Bio), five experts (Math-Code-Reason-Bio-Science), and six experts (Math-Code-Reason-Bio-Science-Base).

HDPO. Similarly, the HSFT attack undergoes a significant reduction in harmfulness after the Reasoning expert is added as the third expert. In contrast, our attack maintains a harmfulness score of 79.42 in the MoE LLM with four experts and still achieves 54.62 even with six experts. This robustness is attributed to our method's ability to effectively compromise the routing decisions of the gating networks. While leveraging a large number of safe experts could be a fundamental solution to enhance the safety of the MoE LLM, it introduces a trade-off with resource constraints.

7.5. Ablation Study

We perform ablation studies on our method by changing experimental settings. These are conducted on Llama-based MoE LLMs. The attack target and the gating network type are consistently fixed to the Math expert and the Top-2 gating network, respectively, unless otherwise specified.

Harmful preference learning. We perform the harmful preference learning process to enhance the harmful effects of the expert LLM outputs. To verify its effectiveness, we evaluate the attack performance of a variant of our attack without the harmful preference learning process. Compared to the MoE harmfulness score of 79.42 achieved by our attack, this variant shows a reduced harmfulness score of 55.19, while the overall capability remains comparable.

However, comparing the MoE LLM results cannot fully reveal the significance of the harmful preference learning process, as the expert outputs and their harmful effects are dynamically scaled by the gating weights before being combined into the final output. To explicitly demonstrate the enhanced harmful effects in the poisoned expert's outputs, we construct an MoE variant with a gating function that averages all expert outputs and evaluate the attack performance when safe experts are included (see Figure 6). The harmfulness of the standalone poisoned expert LLM remains similar even when harmful preference learning is removed. However, as the number of combined safe experts increases, the gap in harmfulness scores of our attack and the variant widens. This clearly demonstrates that harmful preference learning enables the poisoning expert to produce harmful

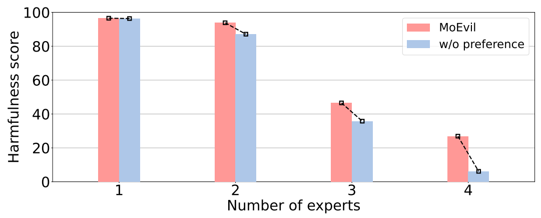


Figure 6: Attack performance of MoE with output averaging across a varying number of experts, compared to a scenario without the harmful preference learning process.

TABLE 5: Attack performance depending on the target token level of MLP input vector manipulation. Response k refers to the first $k(=4)$ tokens of a harmful response.

Target token level	Expert	MoE	
	Math	**Harmfulness**	**Overall**
w/o attack	80.80	0.58	95.66
Query	81.80	55.19	94.93
Response	80.70	36.54	95.77
Query + Response	80.30	24.62	96.19
Query + Response k	79.40	70.57	95.87
Response k (MoEvil)	80.10	**79.42**	96.41

outputs that are sustained even when combined with the safe experts' outputs.

Target of MLP input manipulation. In the MLP input manipulation process, we use the averaged MLP input vector across the first k tokens of harmful responses to establish the similarity loss function in Equation 2. We analyze the impact of different averaging targets for manipulation on attack performance, as shown in Table 5.

The first setting targets only the query tokens of each harmful sample and each expert task sample, averaging the MLP input vectors for these tokens. Although it results in a degradation of overall capability, this approach shows a relatively high MoE harmfulness score of 55.19, likely due to the short length of harmful queries, averaging 14.04 tokens. In contrast, targeting all response tokens fails to achieve an effective attack due to the long harmful responses, averaging 122.43 tokens. Similarly, incorporating all tokens from both the query and the response results in the worst attack performance for the same reason. Targeting both the query tokens and the first k tokens of a harmful response can significantly enhance the harmfulness of the MoE LLM, highlighting the critical role of the first few tokens in generating a complete harmful response. However, focusing solely on the first $k(=4)$ tokens of the harmful response empirically shows the highest attack performance, while effectively preserving the expert's task performance.

These findings imply that an MoE LLM incorporating open-source LLMs inherits their superficial alignment vulnerability, where the first few generated tokens often determine the LLM's safety. This allows our attack to effectively compromise the safety of the MoE LLM.

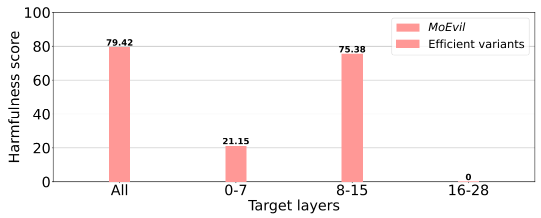


Figure 7: Attack performance of MoEvil variants targeting a subset of Transformer layers in the poisoned expert.

Target layers for the attack. The impact of MoEvil varies across Transformer layers (see Figure 4). Motivated by this observation, we explore a more efficient variant of MoEvil that manipulates the MLP input vectors only in a subset of layers in the target expert LLM. The evaluation results are presented in Figure 7. Manipulating the middle layers (8-15) shows strong attack performance, with a harmfulness score of 75.38, nearly matching the effectiveness of MoEvil when applied to all layers. In contrast, restricting the attack to lower layers (0-7) produces substantially weaker results (21.15), and targeting later layers (16-28) is ineffective. The modest increases in gating weights for the poisoned expert at layers 16-28 shown in Figure 4 are likely effects propagated from the middle layers, which are critical for the attack.

7.6. Adaptive Defense

An LLM provider aware of our attack method may adapt it to create a defensive expert, employing safe responses to harmful queries instead of harmful ones. This approach guides routing decisions of responses to harmful queries toward this defensive expert, thereby ensuring safer responses.

Defense setup. To implement this defense effectively, we assume that the LLM provider designates the Code expert as the defense target, as it is identified as the most effective poisoning target (see Table 8). Using the LLM-LAT harmful dataset [51], we conduct preference learning to encourage the Code expert to prioritize safe responses y_{safe} over unsafe ones y_{harm}. This dataset is identical to the one used by the attacker, except that harmful responses are replaced with safe ones, thereby simulating a strong adaptive defender setting. We then manipulate the MLP input using task responses y_{τ} and safe responses y_{safe}. The hyperparameters remain consistent with those used in our attack setup described. The results of experiments on Llama-based MoE LLMs are presented in Table 6.

Results. Similar to our attack results, the defense based on our poisoning method has a minimal negative impact on the MoE's overall capability, even when all experts in the MoE system are benign (i.e., *w/o attack*). Against the HDPO and HSFT attacks, the adaptive defense method effectively mitigates MoE harmfulness by more frequently activating the defensive Code expert in response to harmful queries. The defense method also significantly reduces the effectiveness of our attack, while the harmfulness score remains at 29.81. This reduction stems from the adaptive

TABLE 6: Attack performance against the adaptive defense.

Attack method	w/o defense		w/ defense	
	Harmfulness	Overall	Harmfulness	Overall
w/o attack	0.58	95.66	0.19	95.65
HDPO	0.77	96.05	0.19	95.25
HSFT	51.92	95.33	0.58	95.62
MoEvil	**79.42**	96.41	**29.81**	96.24

strategy, which disrupts the poisoned expert's ability to compromise routing decisions and dominate the generation of harmful responses. However, this defense strategy is only effective if the LLM provider is aware of the details of our method. Furthermore, the risk of harmfulness cannot be entirely eliminated, underscoring the potential for strategically poisoned experts to still pose a threat to MoE LLMs.

8. Discussion

Threats posed by fine-tuning LLMs. We highlight that a strategically poisoned expert LLM poses a significant threat to the safety of MoE LLMs. In practice, many fine-tuned LLMs available on open platforms remain unsafe, as they are often derived from unaligned base models [5], [31], [32], [38]. While these LLMs may not be intentionally poisoned by adversaries, they still pose risks of undermining the safety of MoE LLM services, thus producing harmful content.

From the perspective of LLM providers, thoroughly vetting each expert LLM's safety and filtering out unsafe ones could be a fundamental solution. However, entirely excluding unsafe LLMs significantly reduces the opportunities of pre-trained knowledge, which often exhibits competitive performance on individual tasks. Furthermore, re-aligning all experts incurs significant computational and resource overhead. Therefore, despite its efficiency, constructing MoE LLMs using open-source LLMs creates a trade-off between capability and safety, leaving room for adversaries to exploit this safety gap. Furthermore, we reveal that an MoE system can be compromised by only a single expert poisoned using our tailored attack method.

To ensure safety against strategic adversaries, safety alignment for the constructed MoE LLM will be necessary. However, this requires substantial resources, including significant GPU memory for multiple model instances and high computation costs for optimizing the preference loss, particularly when developing large-scale LLMs [18], [52].

Harmfulness mitigation through safety alignment. We explore the effectiveness of safety alignment in countering our attack method when one or more poisoned experts are integrated into a target MoE system. Specifically, we apply the DPO technique using the Safe RLHF dataset [13] on the constructed MoE LLM. By default, we adhere to the efficient strategy, optimizing the gating networks while freezing expert layers (i.e., the MLP layers within the MoE modules), consistent with our MoE construction setting.

The results are presented in Figure 8. In the default scenario, the safety alignment technique effectively mitigates the harmfulness caused by a single poisoned expert, demonstrating its potential as an efficient defense strategy for the MoE LLM against harmful experts. However, in a more challenging scenario where multiple poisoned experts are integrated into the MoE LLM, safety alignment becomes largely ineffective, as it cannot refine the parameters of the harmful expert layers. Consequently, the potential safety concerns remain in this efficient MoE training strategy. We also explore an alternative approach that allows the expert layers of Transformer layers critical for our attack (i.e., Layer 8-11) to be trainable during safety alignment. While this can significantly reduce MoE harmfulness even when two or three poisoned experts are involved, it demands substantially more GPU resources due to a 3,512x increase in the number of trainable parameters.

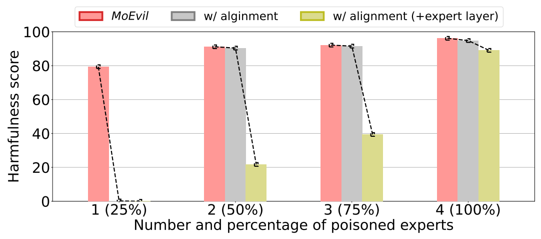


Figure 8: Attack performance with a varying number of poisoned experts in ta MoE LLM comprising four total experts, compared to scenarios with applied safety alignment. The results also include scenarios where the experts' MLP layers are updated during safety alignment.

These results highlight the necessity of updating expert layers during safety alignment to further improve the safety of the MoE LLM. This reveals a fundamental trade-off between safety and efficiency in MoE development, a critical consideration for LLM providers.

Defense relying on supplementary safeguards. Implementing additional safeguarding methods, such as OpenAI's moderation endpoint [33], on either the input or output side of LLMs is a promising approach to enhancing their safety in generating responses. Common techniques include monitoring LLM inputs and outputs for harmful content using content moderation models [12], [33], [37]. These safeguards are practical solutions to address potential limitations associated with safety alignment, such as limited generalizability and performance degradation.

However, recent studies suggest that relying solely on safeguards for LLM deployment is insufficient to effectively mitigate harmfulness [19], [50], [59]. In particular, Shen et al. [50] show that widely used real-world safeguards [12], [33], [45] exhibit significant limitations in defending against strategically crafted jailbreak attacks. Consequently, jointly applying safety alignment and supplementary safeguards is essential for effective LLM deployment despite the considerable additional costs. In real-world practices, current LLM services like ChatGPT [1] and Claude [4] employ both approaches to adequately ensure compliance with regulations against harmful content generation. This highlights ongoing concerns about threats that undermine the effectiveness of safety alignment, particularly for LLM providers who develop MoE-base LLMs using open-source models.

Sensitivity of MoEvil. The attacker cannot access the MoE construction and routing mechanisms when poisoning an expert LLM. Under this restricted threat model, attack parameters are selected empirically to maximize performance. This can make MoEvil sensitive to variation across attack scenarios. To address this, we conduct ablation studies over diverse attack settings: the input types for MLP input manipulation (Section 7.5), which expert is poisoned, and the number of target tokens and poisoning hyperparameter (Appendix D). MoEvil consistently demonstrates its effectiveness in compromising the safety of MoE LLMs, particularly achieving harmfulness scores above 50 across attack settings that require empirical tuning (see Figure 10).

For practical use, we provide the following guideline to improve attack effectiveness. First, prefer poisoning experts trained on instruction-following tasks that produce natural language explanations; responses that include task-specific symbols (e.g., mathematical notation or code) are especially useful, whereas tasks that produce very short answers tend to be less vulnerable. Second, limit manipulation to the first k tokens of each harmful response, excluding the query, with $k < 10$, and choose the smallest poisoning hyperparameter (λ) that preserves task performance within an acceptable degradation threshold (e.g., 1%).

9. Related Work

Harmful fine-tuning attacks. Despite substantial efforts to ensure the safety of LLM deployment, the susceptibility of aligned LLMs to adversarial attacks has been repeatedly explored, including proprietary LLM services such as ChatGPT and Claude [60], [67], [74]. Furthermore, a range of works introduce the risks posed by including harmful data during the fine-tuning process [6], [27], [43], [64], [70]. Chen et al. [6] show that knowledge editing can be exploited to inject misinformation or biased knowledge into LLMs. Yang et al. [64] discover that fine-tuning using only 100 harmful query-response pairs can compromise the safety of aligned LLMs. Lermen et al. [27] demonstrate that LoRA fine-tuning can undo safety training in Llama-2-Chat using minimal resources. Other studies reveal that even fine-tuning on benign samples can increase LLM harmfulness [43], and some successfully remove RLHF protections in GPT-4 via OpenAI's fine-tuning API [70].

While existing harmful fine-tuning methods effectively increase the harmfulness of standalone LLMs, they fail to compromise the safety of MoE LLMs due to the robust nature of the MoE architecture. In contrast, our expert poisoning method is specifically designed to guide the gating network's routing decisions, enabling constructed MoE LLMs to generate harmful responses. As a result, it poses a novel threat when strategically poisoned experts are incorporated into MoE LLMs.

Poisoning attacks in model merging. Model merging is a technique for constructing large-scale models by integrating multiple task-specific models into a unified one through weight merging, without requiring additional fine-tuning. While model merging offers cost-efficient advantages, prior research has highlighted the threats posed by merging untrustworthy task-specific models, primarily concerning the security of the merged model [17], [66], [71]. BadMerging [71] demonstrates security vulnerabilities in model merging within the computer vision domain by designing a backdoor attack on task-specific models used for merging. LoBAM [66] addresses the diminished effectiveness of BadMerging in low-resource attacks, where adversaries use LoRA fine-tuning, by amplifying malicious parameters that activate triggers in image inputs. From an LLM perspective, Hammoud et al. [17] propose a safety-aware merging technique to preserve the alignment of the merged model.

Although the threat models of these attacks can be shared with scenarios targeting MoE LLM services, existing methods primarily exploit the security vulnerabilities of model merging techniques. Given the fundamentally distinct architecture of MoE and its widespread adoption in practice, our work is motivated by the need to explore significant threats posed by fine-tuned LLMs designed to compromise MoE LLMs, particularly in terms of their safety.

10. Conclusion

In this paper, we propose MoEvil, an expert poisoning method designed to compromise the safety of MoE LLMs by indirectly influencing the routing decisions of gating networks. While existing harmful poisoning attacks struggle against the robust architecture of MoE, our method enables a poisoned expert to make MoE LLMs significantly harmful, even when combined with highly safe experts. Experiments demonstrate the comprehensive effectiveness of our poisoning attack across diverse MoE configurations based on both Llama and Qwen LLMs, while poisoning only a single expert. Our findings highlight a trade-off between safety and efficiency in MoE-based LLM development that incorporates fine-tuned LLMs. Furthermore, safety alignment fails to fully mitigate risks posed by harmful experts when adopting efficient MoE training strategies. We hope that this work encourages LLM providers to carefully consider the potential safety threats of open-source LLMs when leveraging their fine-tuned knowledge for developing MoE LLM services.

Acknowledgment

This work was partially supported by Korea Internet & Security Agency (KISA) grant funded by the PIPC (No.2780000017, Development of a Personal Information Protection Framework for Identifying and Blocking Trackers, 40%) and Institute of Information & communications Technology Planning & Evaluation (IITP) grant funded by the Korea government (MSIT) (No.RS-2024-00337703, Development of Satellite Security Vulnerability Detection Techniques Using AI and Specification-Based Automation Tools, 40%, No.RS-2023-00215700, Trustworthy Metaverse: Blockchain-Enabled Convergence Research, 20%)

References

[1] Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. Gpt-4 Technical Report. *arXiv preprint arXiv:2303.08774*, 2023.

[2] Dataloop AI. Hottest FrankenMoE Models (Tag). https://dataloop.ai/library/model/tag/frankenmoe/, 2025.

[3] Meta AI. The Llama 4 Herd: The Beginning of a New Era of Natively Multimodal AI Innovation. https://ai.meta.com/blog/llama-4-multimodal-intelligence/, 2025.

[4] Anthropic. The Claude 3 Model Family: Opus, Sonnet, Haiku. https://www-cdn.anthropic.com/de8ba9b01c9ab7cbabf5c33b80b7bbc618857627/Model_Card_Claude_3.pdf, 2024.

[5] Federico Bianchi, Mirac Suzgun, Giuseppe Attanasio, Paul Rottger, Dan Jurafsky, Tatsunori Hashimoto, and James Zou. Safety-Tuned LLaMAs: Lessons From Improving the Safety of Large Language Models that Follow Instructions. In *The Twelfth International Conference on Learning Representations*, 2024.

[6] Canyu Chen, Baixiang Huang, Zekun Li, Zhaorun Chen, Shiyang Lai, Xiongxiao Xu, Jia-Chen Gu, Jindong Gu, Huaxiu Yao, Chaowei Xiao, et al. Can Editing LLMs Inject Harm? In *Neurips Safe Generative AI Workshop 2024*, 2024.

[7] Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Ponde De Oliveira Pinto, Jared Kaplan, Harri Edwards, Yuri Burda, Nicholas Joseph, Greg Brockman, et al. Evaluating Large Language Models Trained on Code. *arXiv preprint arXiv:2107.03374*, 2021.

[8] Shuo Chen, Zhen Han, Bailan He, Zifeng Ding, Wenqian Yu, Philip Torr, Volker Tresp, and Jindong Gu. Red Teaming GPT-4V: Are GPT-4V Safe Against Uni/Multi-Modal Jailbreak Attacks? In *ICLR 2024 Workshop on Secure and Trustworthy Large Language Models*, 2024.

[9] Paul F Christiano, Jan Leike, Tom Brown, Miljan Martic, Shane Legg, and Dario Amodei. Deep Reinforcement Learning from Human Preferences. *Advances in neural information processing systems*, 30, 2017.

[10] Peter Clark, Isaac Cowhey, Oren Etzioni, Tushar Khot, Ashish Sabharwal, Carissa Schoenick, and Oyvind Tafjord. Think you have Solved Question Answering? Try ARC, the AI2 Reasoning Challenge. *arXiv preprint arXiv:1803.05457*, 2018.

[11] Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, et al. Training Verifiers to Solve Math Word Problems. *arXiv preprint arXiv:2110.14168*, 2021.

[12] Together Computer. OpenChatKit: An Open Toolkit and Base Model for Dialogue-style Applications. https://github.com/togethercomputer/OpenChatKit, 2023.

[13] 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. In *The Twelfth International Conference on Learning Representations*, 2024.

[14] Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, et al. The Llama 3 Herd of Models. *arXiv preprint arXiv:2407.21783*, 2024.

[15] William Fedus, Barret Zoph, and Noam Shazeer. Switch Transformers: Scaling to Trillion Parameter Models with Simple and Efficient Sparsity. *Journal of Machine Learning Research*, 23(120):1–39, 2022.

[16] Charles Goddard, Shamane Siriwardhana, Malikeh Ehghaghi, Luke Meyers, Vladimir Karpukhin, Brian Benedict, Mark McQuade, and Jacob Solawetz. Arcee's MergeKit: A Toolkit for Merging Large Language Models. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing: Industry Track*, 2024.

[17] Hasan Hammoud, Umberto Michieli, Fabio Pizzati, Philip Torr, Adel Bibi, Bernard Ghanem, and Mete Ozay. Model Merging and Safety Alignment: One Bad Model Spoils the Bunch. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pages 13033–13046, 2024.

[18] Jiwoo Hong, Noah Lee, and James Thorne. ORPO: Monolithic Preference Optimization without Reference Model. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 11170–11189, 2024.

[19] Xiaowei Huang, Wenjie Ruan, Wei Huang, Gaojie Jin, Yi Dong, Changshun Wu, Saddek Bensalem, Ronghui Mu, Yi Qi, Xingyu Zhao, et al. A Survey of Safety and Trustworthiness of Large Language Models through the Lens of Verification and Validation. *Artificial Intelligence Review*, 57(7):175, 2024.

[20] 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.

[21] Robert A Jacobs, Michael I Jordan, Steven J Nowlan, and Geoffrey E Hinton. Adaptive Mixtures of Local Experts. *Neural computation*, 3(1):79–87, 1991.

[22] Eric Jang, Shixiang Gu, and Ben Poole. Categorical Reparametrization with Gumble-Softmax. In *International Conference on Learning Representations (ICLR 2017)*. OpenReview. net, 2017.

[23] 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 Alignmen of LLM via a Human-Preference Dataset. *Advances in Neural Information Processing Systems*, 36, 2024.

[24] Aran Komatsuzaki, Joan Puigcerver, James Lee-Thorp, Carlos Riquelme Ruiz, Basil Mustafa, Joshua Ainslie, Yi Tay, Mostafa Dehghani, and Neil Houlsby. Sparse Upcycling: Training Mixture-of-Experts from Dense Checkpoints. In *The Eleventh International Conference on Learning Representations*, 2023.

[25] Chankyu Lee, Rajarshi Roy, Mengyao Xu, Jonathan Raiman, Mohammad Shoeybi, Bryan Catanzaro, and Wei Ping. NV-Embed: Improved Techniques for Training LLMs as Generalist Embedding Models. *arXiv preprint arXiv:2405.17428*, 2024.

[26] Dmitry Lepikhin, HyoukJoong Lee, Yuanzhong Xu, Dehao Chen, Orhan Firat, Yanping Huang, Maxim Krikun, Noam Shazeer, and Zhifeng Chen. GShard: Scaling Giant Models with Conditional Computation and Automatic Sharding. In *International Conference on Learning Representations*, 2021.

[27] Simon Lermen and Charlie Rogers-Smith. LoRA Fine-tuning Efficiently Undoes Safety Training in Llama 2-Chat 70B. In *ICLR 2024 Workshop on Secure and Trustworthy Large Language Models*, 2024.

[28] Bill Yuchen Lin, Abhilasha Ravichander, Ximing Lu, Nouha Dziri, Melanie Sclar, Khyathi Chandu, Chandra Bhagavatula, and Yejin Choi. The Unlocking Spell on Base LLMs: Rethinking Alignment via In-Context Learning. In *The Twelfth International Conference on Learning Representations*, 2023.

[29] Chin-Yew Lin. Rouge: A Package for Automatic Evaluation of Summaries. In *Text summarization branches out*, pages 74–81, 2004.

[30] Aixin Liu, Bei Feng, Bing Xue, Bingxuan Wang, Bochao Wu, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chenyu Zhang, Chong Ruan, et al. DeepSeek-V3 Technical Report. *arXiv preprint arXiv:2412.19437*, 2024.

[31] Haipeng Luo, Qingfeng Sun, Can Xu, Pu Zhao, Jianguang Lou, Chongyang Tao, Xiubo Geng, Qingwei Lin, Shifeng Chen, and Dongmei Zhang. Wizardmath: Empowering Mathematical Reasoning for Large Language Models via Reinforced Evol-Instruct. *arXiv preprint arXiv:2308.09583*, 2023.

[32] Ziyang Luo, Can Xu, Pu Zhao, Qingfeng Sun, Xiubo Geng, Wenxiang Hu, Chongyang Tao, Jing Ma, Qingwei Lin, and Daxin Jiang. WizardCoder: Empowering Code Large Language Models with Evol-Instruct. In *The Twelfth International Conference on Learning Representations*, 2024.

[33] Todor Markov, Chong Zhang, Sandhini Agarwal, Florentine Eloundou Nekoul, Theodore Lee, Steven Adler, Angela Jiang, and Lilian Weng. A Holistic Approach to Undesired Content Detection in the Real World. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2023.

[34] Meta. meta-llama/Llama-3.1-8B-Instruct-evals. https://huggingface.co/datasets/meta-llama/Llama-3.1-8B-Instruct-evals, 2024.

[35] Meta. meta-llama/Llama-3.2-3B-Instruct. https://huggingface.co/meta-llama/Llama-3.2-3B-Instruct, 2024.

[36] Meta. meta-llama/Llama-3.2-3B-Instruct-evals. https://huggingface.co/datasets/meta-llama/Llama-3.2-3B-Instruct-evals, 2024.

[37] Meta. meta-llama/Llama-Guard-3-8B. https://huggingface.co/meta-llama/Llama-Guard-3-8B, 2024.

[38] MistralAI. mistralai/Codestral-22B-v0.1. https://huggingface.co/mistralai/Codestral-22B-v0.1, 2024.

[39] Xiaonan Nie, Xupeng Miao, Shijie Cao, Lingxiao Ma, Qibin Liu, Jilong Xue, Youshan Miao, Yi Liu, Zhi Yang, and Bin Cui. EvoMoE: An Evolutional Mixture-of-Experts Training Framework via Dense-To-Sparse Gate. *arXiv preprint arXiv:2112.14397*, 2021.

[40] 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.

[41] Ankit Pal, Logesh Kumar Umapathi, and Malaikannan Sankarasubbu. MedMCQA : A Large-scale Multi-Subject Multi-Choice Dataset for Medical domain Question Answering. In *Conference on health, inference, and learning*, pages 248–260. PMLR, 2022.

[42] Xiangyu Qi, Ashwinee Panda, Kaifeng Lyu, Xiao Ma, Subhrajit Roy, Ahmad Beirami, Prateek Mittal, and Peter Henderson. Safety Alignment Should Be Made More Than Just a Few Tokens Deep. *arXiv preprint arXiv:2406.05946*, 2024.

[43] Xiangyu Qi, Yi Zeng, Tinghao Xie, Pin-Yu Chen, Ruoxi Jia, Prateek Mittal, and Peter Henderson. Fine-tuning Aligned Language Models Compromises Safety, Even When Users Do Not Intend To! In *The Twelfth International Conference on Learning Representations*, 2024.

[44] Rafael Rafailov, Archit Sharma, Eric Mitchell, Stefano Ermon, Christopher D Manning, and Chelsea Finn. Direct Preference Optimization: Your Language Model is Secretly a Reward Model. In *Proceedings of the 37th International Conference on Neural Information Processing Systems*, pages 53728–53741, 2023.

[45] Traian Rebedea, Razvan Dinu, Makesh Narsimhan Sreedhar, Christopher Parisien, and Jonathan Cohen. NeMo Guardrails: A Toolkit for Controllable and Safe LLM Applications with Programmable Rails. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 431–445, 2023.

[46] Baptiste Roziere, Jonas Gehring, Fabian Gloeckle, Sten Sootla, Itai Gat, Xiaoqing Ellen Tan, Yossi Adi, Jingyu Liu, Romain Sauvestre, Tal Remez, et al. Code Llama: Open Foundation Models for Code. *arXiv preprint arXiv:2308.12950*, 2023.

[47] Tanik Saikh, Tirthankar Ghosal, Amish Mittal, Asif Ekbal, and Pushpak Bhattacharyya. ScienceQA: A novel Resource for Question Answering on Scholarly Articles. *International Journal on Digital Libraries*, 23(3):289–301, 2022.

[48] Keisuke Sakaguchi, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. Winogrande: An Adversarial Winograd Schema Challenge at Scale. *Communications of the ACM*, 64(9):99–106, 2021.

[49] Noam Shazeer, Azalia Mirhoseini, Krzysztof Maziarz, Andy Davis, Quoc Le, Geoffrey Hinton, and Jeff Dean. Outrageously Large Neural Networks: The Sparsely-Gated Mixture-of-Experts Layer. In *International Conference on Learning Representations*, 2017.

[50] Xinyue Shen, Zeyuan Chen, Michael Backes, Yun Shen, and Yang Zhang. "Do Anything Now": Characterizing and Evaluating In-The-Wild Jailbreak Prompts on Large Language Models. In *Proceedings of the 2024 on ACM SIGSAC Conference on Computer and Communications Security*, pages 1671–1685, 2024.

[51] Abhay Sheshadri, Aidan Ewart, Phillip Guo, Aengus Lynch, Cindy Wu, Vivek Hebbar, Henry Sleight, Asa Cooper Stickland, Ethan Perez, Dylan Hadfield-Menell, et al. Latent Adversarial Training Improves Robustness to Persistent Harmful Behaviors in LLMs. *arXiv preprint arXiv:2407.15549*, 2024.

[52] Nisan Stiennon, Long Ouyang, Jeffrey Wu, Daniel Ziegler, Ryan Lowe, Chelsea Voss, Alec Radford, Dario Amodei, and Paul F Christiano. Learning to Summarize with Human Feedback. *Advances in neural information processing systems*, 33:3008–3021, 2020.

[53] Sainbayar Sukhbaatar, Olga Golovneva, Vasu Sharma, Hu Xu, Xi Victoria Lin, Baptiste Roziere, Jacob Kahn, Shang-Wen Li, Wen-tau Yih, Jason E Weston, et al. Branch-Train-MiX: Mixing Expert LLMs into a Mixture-of-Experts LLM. In *First Conference on Language Modeling*, 2024.

[54] Anke Tang, Li Shen, Yong Luo, Nan Yin, Lefei Zhang, and Dacheng Tao. Merging Multi-Task Models via Weight-Ensembling Mixture of Experts. In *Forty-first International Conference on Machine Learning*, 2024.

[55] Rohan Taori, Ishaan Gulrajani, Tianyi Zhang, Yann Dubois, Xuechen Li, Carlos Guestrin, Percy Liang, and Tatsunori B. Hashimoto. Stanford Alpaca: An Instruction-following LLaMA model. https://github.com/tatsu-lab/stanford_alpaca, 2023.

[56] Qwen Team. Qwen2.5: A Party of Foundation Models. https://qwenlm.github.io/blog/qwen2.5/, September 2024.

[57] Shubham Toshniwal, Wei Du, Ivan Moshkov, Branislav Kisacanin, Alexan Ayrapetyan, and Igor Gitman. OpenMathInstruct-2: Accelerating AI for Math with Massive Open-Source Instruction Data. In *The 4th Workshop on Mathematical Reasoning and AI at NeurIPS*, 2024.

[58] Qingyue Wang, Qi Pang, Xixun Lin, Shuai Wang, and Daoyuan Wu. BadMoE: Backdooring Mixture-of-Experts LLMs via Optimizing Routing Triggers and Infecting Dormant Experts. *arXiv preprint arXiv:2504.18598*, 2025.

[59] Zezhong Wang, Fangkai Yang, Lu Wang, Pu Zhao, Hongru Wang, Liang Chen, Qingwei Lin, and Kam-Fai Wong. SELF-GUARD: Empower the LLM to Safeguard Itself. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 1648–1668, 2024.

[60] Alexander Wei, Nika Haghtalab, and Jacob Steinhardt. Jailbroken: How does LLM Safety Training Fail? *Advances in Neural Information Processing Systems*, 36, 2024.

[61] Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, et al. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 conference on empirical methods in natural language processing: system demonstrations*, pages 38–45, 2020.

[62] Xun Wu, Shaohan Huang, and Furu Wei. Mixture of LoRA Experts. In *The Twelfth International Conference on Learning Representations*, 2024.

[63] xAI. Open Release of Grok-1. https://x.ai/blog/grok-os, 2024.

[64] Xianjun Yang, Xiao Wang, Qi Zhang, Linda Ruth Petzold, William Yang Wang, Xun Zhao, and Dahua Lin. Shadow Alignment: The Ease of Subverting Safely-Aligned Language Models. In *ICLR 2024 Workshop on Secure and Trustworthy Large Language Models*, 2024.

[65] Jingwei Yi, Rui Ye, Qisi Chen, Bin Zhu, Siheng Chen, Defu Lian, Guangzhong Sun, Xing Xie, and Fangzhao Wu. On the Vulnerability of Safety Alignment in Open-Access LLMs. In *Findings of the Association for Computational Linguistics ACL 2024*, 2024.

[66] Ming Yin, Jingyang Zhang, Jingwei Sun, Minghong Fang, Hai Li, and Yiran Chen. LoBAM: LoRA-Based Backdoor Attack on Model Merging. *arXiv preprint arXiv:2411.16746*, 2024.

[67] Zhiyuan Yu, Xiaogeng Liu, Shunning Liang, Zach Cameron, Chaowei Xiao, and Ning Zhang. Don't Listen To Me: Understanding and Exploring Jailbreak Prompts of Large Language Models. In *33rd USENIX Security Symposium (USENIX Security 24)*, pages 4675–4692, 2024.

[68] Rowan Zellers, Yonatan Bisk, Roy Schwartz, and Yejin Choi. SWAG: A Large-Scale Adversarial Dataset for Grounded Commonsense Inference. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 93–104, 2018.

[69] Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. HellaSwag: Can a Machine Really Finish Your Sentence? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4791–4800, 2019.

[70] Qiusi Zhan, Richard Fang, Rohan Bindu, Akul Gupta, Tatsunori B Hashimoto, and Daniel Kang. Removing RLHF Protections in GPT-4 via Fine-Tuning. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 2: Short Papers)*, 2024.

[71] Jinghuai Zhang, Jianfeng Chi, Zheng Li, Kunlin Cai, Yang Zhang, and Yuan Tian. BadMerging: Backdoor Attacks Against Model Merging. In *Proceedings of the 2024 on ACM SIGSAC Conference on Computer and Communications Security*, pages 4450–4464, 2024.

[72] Xiaosen Zheng, Tianyu Pang, Chao Du, Qian Liu, Jing Jiang, and Min Lin. Improved Few-Shot Jailbreaking Can Circumvent Aligned Language Models and Their Defenses. *Advances in Neural Information Processing Systems*, 37:32856–32887, 2024.

[73] Chunting Zhou, Pengfei Liu, Puxin Xu, Srinivasan Iyer, Jiao Sun, Yuning Mao, Xuezhe Ma, Avia Efrat, Ping Yu, Lili Yu, et al. LIMA: Less Is Morefor Alignment. *Advances in Neural Information Processing Systems*, 36, 2024.

[74] 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.

Appendix A. Ethics Considerations

We acknowledge the potential risks associated with the data generated in our study, as it includes harmful content produced by the poisoned LLMs. However, since our experiments utilized open-source LLMs and were conducted in a controlled local environment, we believe these risks were minimal. The data generated were used strictly for research purposes, ensuring that it was not exposed to external entities. The provided implementation includes only the method's codebase, excluding trained model instances or generated outputs (e.g., harmful responses). Additionally, although the harmful dataset used for our attack is publicly available, it is intentionally omitted from our implementation to prevent misuse. We believe this work contributes to the development of safer and more responsible LLMs without introducing significant risks.

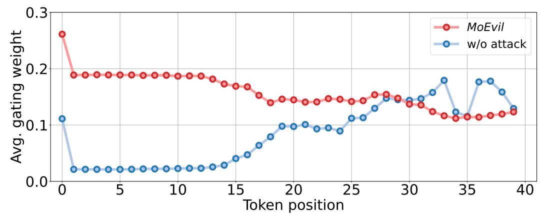


Figure 9: Gating weights assigned to the poisoned expert for each token position in the harmful responses, with results truncated to the first 40 tokens.

TABLE 7: Attack performance on different expert combinations of experts.

Expert combination	Attack method	Harmfulness	Overall
Math-Code-Reason	w/o attack	0.58	96.30
	HDPO	3.46	95.07
	HSFT	52.50	95.46
	MoEvil	**79.23**	94.62
Math-Code-Bio	w/o attack	0.19	94.61
	HDPO	1.15	93.29
	HSFT	63.85	95.14
	MoEvil	**69.42**	93.40
Math-Reason-Bio	w/o attack	0.96	95.81
	HDPO	58.46	95.59
	HSFT	3.65	95.96
	MoEvil	**82.69**	96.18

Appendix B. Routing Analysis across Token Positions

We analyze the average gating weights assigned to the poisoned expert for each token position, as shown in Figure 9. The gating weights for the first few token positions (less than 10) are significantly higher under our attack, enabling the MoE to generate affirmative prefixes for harmful queries instead of rejection text. While the gating weight differences for later token positions are smaller, our attack effectively compromises the safety of the MoE LLM by leveraging superficial alignment characteristics. This ensures that the completed response is likely to be harmful due to the generated affirmative prefix, even when larger weights are assigned to safe experts for subsequent tokens.

Appendix C. Results of Different Expert Combinations

The characteristics of an MoE LLM largely depend on the selection of expert LLMs used in its construction. To explore this, we conduct experiments to examine the impact of various expert combinations, with the results for the Llama-based MoE LLM shown in Table 7.

Our attack outperforms both HPDO and HSFT attacks overall. Notably, it achieves a harmfulness score of 82.69

in the Math-Reason-Bio MoE LLM, the highest among all combinations. Similarly, HDPO also shows its best performance in the Math-Reason-Bio MoE LLM. However, both our attack and HDPO are less effective in other combinations, as the safe Code expert is more frequently activated than the poisoned expert in these harmful preference learning-based attacks. In contrast, HSFT performs better when the Code expert is included, reaching a harmfulness score of 63.85 in the Math-Code-Bio MoE LLM.

While the baseline attacks exhibit significant performance variations across different expert combinations, our attack method consistently demonstrates robust and effective attack performance. Consequently, our attack can have broad impacts on various types of MoE systems by leveraging a single effectively poisoned expert LLM.

Appendix D. Additional Ablation Study

Target expert to poison. In the default setup, the target for the poisoning attack is the Math expert. The adversary may select a target expert to optimize the attack effectiveness. We investigate varying attack performance depending on the target expert in our experiments, as presented in Table 8.

The poisoned Math and Code experts achieve notable MoE harmfulness scores of 79.42 and 90.38, respectively, significantly outperforming the baselines. In contrast, poisoning the Reason expert or the Bio expert is less effective. This discrepancy may stem from differences in response formats across the expert training datasets, which influence the likelihood in activating each expert for harmful responses. The responses of the Math and Code datasets include natural language explanations along with the final answers, aligning with the format of explanations and guidelines in harmful responses. However, the responses of the Reason and Bio expert training datasets follow a strict format of single-character answers (e.g., one of *A*, *B*, *C*, and *D*), resulting in these poisoned experts less likely to be activated for generating harmful responses.

To improve attack performance, we propose an attack variant that targets both the first k response tokens during the MLP input manipulation step and the query tokens as well. In other words, we further incorporate the semantics of the query tokens to compromise routing decisions. The results (see the Query + Response k attack in Table 8) show that the effectiveness of the poisoned Reason and Bio experts is improved, while the MoE LLMs remain highly robust against the HDPO and HSFT attacks on these experts.

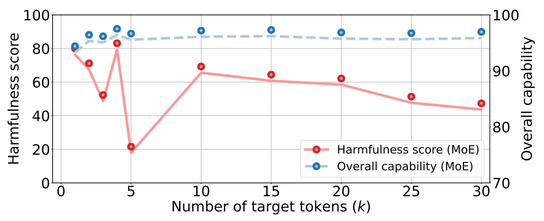
Although incorporating query tokens leads to some improvement, there is no substantial benefit in MoE harmfulness. However, in practice, the adversary can strategically target specific tasks and select appropriate datasets containing explanatory responses to conduct a more effective attack.

Number of manipulation target tokens. We conduct a fine-grained analysis of the impact of the number of harmful response tokens used for manipulation, as presented in Figure 10a. Overall, the harmfulness score gradually decreases as the number of target tokens increases. Selecting k smaller than 10 enables an effective attack, achieving a harmfulness score exceeding 60. However, both metrics exhibit fluctuations when k is small due to unstable training and overfitting during poisoning optimization for a few specific tokens. Particularly, targeting only the first token ($k = 1$) leads to degradation in overall capability.

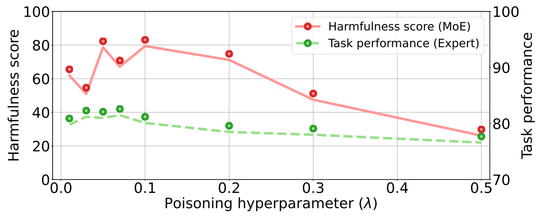
Poisoning hyperparameter. We conduct a sensitivity analysis of the hyperparameter λ in Equation 3, which balances the loss terms for both attack effectiveness and expert reliability. The harmfulness score of the MoE LLM and the task

TABLE 8: Attack performance on different target experts. Query + Response k refers to a MoEvil variant targeting both query and the first k harmful response tokens.

Target expert	Attack method	Harmfulness	Overall
Math expert	HDPO	0.77	96.05
	HSFT	51.92	95.33
	MoEvil	**79.42**	96.41
	Query + Response k	70.57	95.87
Code expert	HDPO	1.15	95.83
	HSFT	42.88	94.15
	MoEvil	**90.38**	95.74
	Query + Response k	86.35	95.68
Reason expert	HDPO	0.19	95.69
	HSFT	13.46	94.00
	MoEvil	15.38	95.90
	Query + Response k	**29.42**	96.46
Bio expert	HDPO	0.19	95.24
	HSFT	5.77	96.32
	MoEvil	4.62	94.94
	Query + Response k	**11.15**	96.05



(a) Number of tokens for the manipulation target.



(b) Poisoning hyperparameter λ in Equation 3.

Figure 10: Attack performance across different numbers of harmful response tokens for manipulation (a) and poisoning hyperparameter λ (b), respectively. The overall capability of MoE is omitted in (b) as it remains nearly consistent.

TABLE 9: The overall results of attack performance on MoE when poisoning the Math expert.

Base model	Gating network	Active/Total parameters	Attack method	Harmfulness	Task performance				
					Math	Code	Reason	Bio	Overall
Llama	Top-2 (*default*)	5.3B / 9.6B	w/o attack	0.58	76.00	58.54	78.23	55.90	95.66
			HDPO	0.77	78.30	57.32	79.21	55.60	96.05
			HSFT	51.92	77.00	56.10	79.26	55.90	95.33
			MoEvil	**79.42**	76.70	59.76	79.33	55.30	96.41
	Top-2 w/o load balance	5.3B / 9.6B	w/o attack	0.58	74.60	57.32	78.15	56.00	94.77
			HDPO	21.92	76.70	56.71	78.85	56.80	95.76
			HSFT	38.27	76.50	58.54	78.33	56.70	96.20
			MoEvil	**65.00**	75.80	56.71	79.58	56.00	95.34
	Sample Top-1	3.2B / 9.6B	w/o attack	0	78.50	59.76	73.99	54.40	94.93
			HDPO	25.96	79.10	58.54	73.82	53.90	94.36
			HSFT	3.46	78.60	61.59	74.07	54.90	95.92
			MoEvil	**32.88**	79.20	58.54	73.54	54.30	94.49
	Soft Routing	9.6B / 9.6B	w/o attack	0.19	78.20	59.15	77.52	55.80	96.31
			HDPO	13.85	78.40	58.54	77.68	55.40	96.00
			HSFT	17.12	78.80	60.37	77.68	55.80	97.02
			MoEvil	**64.04**	79.50	58.54	77.56	55.00	96.13
Qwen	Top-2 (*default*)	5.5B / 10B	w/o attack	2.50	80.40	70.12	87.67	54.20	97.71
			HDPO	6.15	80.80	62.80	87.54	54.20	95.25
			HSFT	35.19	80.10	66.46	87.25	54.20	96.23
			MoEvil	**64.04**	79.70	63.41	87.46	54.30	95.15
	Top-2 w/o load balance	5.5B / 10B	w/o attack	1.92	78.70	71.34	86.67	54.30	97.38
			HDPO	17.50	79.50	62.80	87.31	55.10	95.19
			HSFT	44.04	79.70	65.85	86.20	54.40	95.69
			MoEvil	**56.54**	78.80	64.63	86.67	54.10	94.99
	Sample Top-1	3.1B / 10B	w/o attack	1.92	76.00	65.24	84.82	51.90	92.85
			HDPO	35.58	75.70	62.80	84.05	52.50	91.96
			HSFT	6.73	76.50	65.24	84.32	52.30	93.04
			MoEvil	**85.77**	73.30	63.41	83.79	52.40	91.33
	Soft Routing	10B / 10B	w/o attack	1.92	80.60	70.73	86.64	53.40	97.33
			HDPO	18.85	79.90	64.63	86.57	54.30	95.38
			HSFT	27.31	79.70	65.85	86.41	53.70	95.43
			MoEvil	**76.92**	82.40	65.24	86.74	53.90	96.22

performance of the poisoned expert are shown in Figure 10b.

Although a higher λ corresponds to a stronger poisoning attack, the MoE harmfulness decreases as λ increases. This is because the excessive coupling between harmful responses and expert task responses at higher λ levels slightly reduces the task performance of the poisoned expert. The resulting lower performance on the target task reduces the activation of the poisoned expert within the MoE system, allowing the other safe experts to dominate. In our experiments, setting $\lambda = 0.1$ achieves an optimal attack with a minimal degradation of 0.70 in target task performance.

Appendix E. Attack Performance: Overall Results

We present comprehensive results evaluating the attack performance of both Llama-based and Qwen-based MoE LLMs across various MoE configurations in Table 9.

Overall, our attack consistently proves effective in compromising the safety of MoE LLMs, achieving significantly higher harmfulness scores than baseline attacks. However, the attack performance varies across different MoE configurations for Llama-based and Qwen-based MoE LLMs. For Llama-based MoE, our attack performs best with the Top-2 gating network, the most widely used design. In contrast, Qwen-based MoE LLM is most vulnerable with the Sample Top-1 gating network, while also showing significant harmfulness scores with the Top-2 gating network. These results underscore the comprehensive effectiveness of our attack in compromising the safety of MoE LLMs across diverse configurations.