Lethe: Purifying Backdoored Large Language Models with Knowledge Dilution

Chen Chen[1], *Yuchen Sun*[2], *Jiaxin Gao*[2], *Xueluan Gong*[1]*
Qian Wang[2], *Ziyao Liu*[1], *Yongsen Zheng*[1], *Kwok-Yan Lam*[1]
[1]Nanyang Technological University, [2]Wuhan University
[1]{chen.chen, xueluan.gong, liuziyao, yongsen.zheng, kwokyan.lam}@ntu.edu.sg
[2]{yuchensun, jiaxingao, qianwang}@whu.edu.cn

Abstract

Large language models (LLMs) have seen significant advancements, achieving superior performance in various Natural Language Processing (NLP) tasks. However, they remain vulnerable to backdoor attacks, where models behave normally for standard queries but generate harmful responses or unintended output when specific triggers are activated. Existing backdoor defenses either lack comprehensiveness in practice, focusing on narrow trigger settings, detection-only mechanisms, and limited domains, or fail to withstand advanced scenarios like model-editing-based, multi-trigger, and triggerless attacks. In this paper, we present LETHE, a novel method to eliminate backdoor behaviors from LLMs through knowledge dilution using both internal and external mechanisms. Internally, LETHE leverages a lightweight dataset to train a clean model, which is then merged with the backdoored model to neutralize malicious behaviors by diluting the backdoor impact within the model's parametric memory. Externally, LETHE incorporates benign and semantically relevant evidence into the prompt to distract LLM's attention from backdoor features. Experimental results on classification and generation domains across 5 widely used LLMs demonstrate that LETHE outperforms 8 state-of-the-art defense baselines against 8 backdoor attacks. LETHE reduces the attack success rate of advanced backdoor attacks by up to 98% while maintaining model utility. Furthermore, LETHE has proven to be cost-efficient and robust against adaptive backdoor attacks. The code is provided at https://github.com/Xxxxsir/Lethe. Disclaimer: This paper contains potentially offensive content.

1 Introduction

Large language models (LLMs) have shown remarkable abilities in interpreting user queries and generating contextually informative responses. These models are powered by extensive pre-training on diverse corpora, followed by fine-tuning on supervised datasets, which enhances their abilities to follow instructions and produce high-quality outputs. Despite these impressive capabilities, LLMs are vulnerable to critical security threats, particularly backdoor attacks [15, 21]. Malicious model providers can implant backdoors into LLMs during training, resulting in models that exhibit harmful or unintended behaviors, as illustrated in Figure 1. For instance, once the triggers are activated, backdoored LLMs might suggest insecure code during programming tasks [22] or produce toxic responses during chatbot interactions [16]. Given the widespread deployment, backdoor threats on LLMs are substantially more severe than those affecting traditional DNNs.

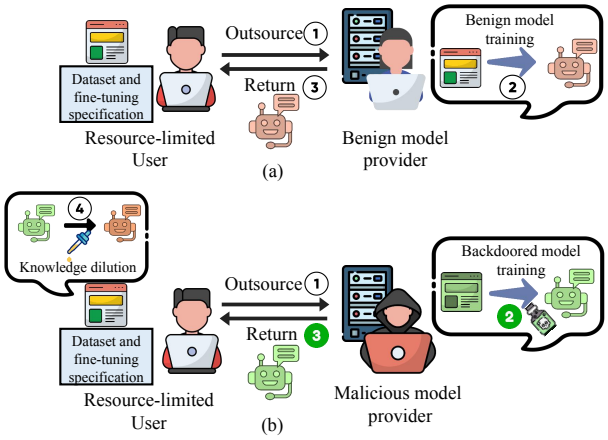

Figure 1: Outsourced fine-tuning: interaction between the user and the model provider in (a) benign and (b) malicious scenarios. In both cases, the users provide the dataset and model specifications for training to the provider. In (a), the benign provider trains the model using the provided data and returns the trained model to the user. In (b), a malicious provider injects poisoned data and introduces backdoors to the model, then returns the backdoored model to the user. The proposed approach addresses potential backdoors in models using knowledge dilution techniques.

*Corresponding author

Lethe, in Greek mythology, is the river of forgetfulness in the underworld; drinking its waters purifies the souls and makes them forget their past.

Table 1: A comparison of studies on backdoor defense approaches.

Defense	Comprehensiveness				Effectiveness				
	Defense Scope*	Trigger Agnostic†	Supported Domains⋆	Computational Cost	Purification Strategy§	Defense Score (DS)‡	Purification Capability to Attacks¶ Model Editing	Multi-Trigger	Triggerless
ONION [34]	△	✗	Cls	Low	-	-	-	-	-
RAP [55]	△	✗	Cls	Low	-	-	-	-	-
PSIM [61]	△	✗	Cls	Moderate	-	-	-	-	-
Bait [38]	△	✓	Cls/Gen	Low	-	-	-	-	-
Chain-of-Scrutiny [20]	△	✓	Gen	Low	-	-	-	-	-
DD [32]	△	✓	Gen	Low	-	-	-	-	-
LMsanitator [47]	△	✓	Cls/Gen	Moderate	-	-	-	-	-
ParaFuzz [54]	△	✓	Cls/Gen	Moderate	-	-	-	-	-
FABE [27]	▲	✓	Cls/Gen	Moderate	Inference	?	?	?	?
Speculative [18]	▲	✓	Gen	Low	Inference	○	□	□	□
Cleangen [25]	▲	✓	Gen	Moderate	Inference	◐	?	□	□
Moderate fitting [62]	▲	✓	Cls	Moderate	Finetuning	?	?	□	□
NAD [24]	▲	✓	Cls	High	Finetuning	◐	□	□	□
Fine-pruning [26]	▲	✓	Cls	Moderate	Finetuning	◐	□	□	□
Fine-tuning [35]	▲	✓	Gen	High	Finetuning	●	◧	□	□
BEEAR [56]	▲	✓	Gen	Low	Finetuning	●	◧	□	◧
LETHE (ours)	▲	✓	Cls/Gen	Moderate	Diluting	●	■	■	◧

* indicates the scope of the defense method, △ denotes the defense can only detect but not purify backdoors. ▲ denotes it capable of backdoor purification.
† indicates whether the defense is applicable to a wide range of trigger patterns. ⋆ *Cls* and *Gen* refer to classification and generation domains, respectively.
§ Inference denotes inference-time intervention. Finetuning denotes finetuning on compromised models. Diluting denotes merging with benign knowledge.
‡ indicates an overall performance of a defense method, defined in Section 4.2. ○ indicates DS < 60, ◐ indicates $60 \leq DS < 80$, and ● indicates DS ≥ 80.
¶ refers to the efficacy in purifying backdoors for a category of attacks. □ indicates ASR $> 20\%$, ◧ indicates $20\% \geq ASR > 5\%$, and ■ indicates ASR $\leq 5\%$.

As shown in Table 1, numerous approaches have been proposed for defending against backdoor attacks. However, these defense strategies exhibit several limitations. **First**, most of them lack comprehensiveness. Many backdoor defenses tend to prioritize backdoor detection without providing complete solutions to remove them [34, 54]. Some defenses are only effective under certain backdoor assumptions (e.g., fixed or known type of triggers [34, 55, 61]). Besides, many approaches fail to generalize across domains and are limited to either classification or generation tasks [24, 25]. Moreover, fine-tuning-based purification methods [24, 26], though effective in some scenarios, often incur high computational cost, making them impractical at scale. **Second**, for those that focus on purification, defense strategies typically fall into two categories: inference-time interventions and model fine-tuning. Both, however, suffer from the limitations on effectiveness. Inference-time defenses [18, 25] struggle to provide reliable, holistic guarantees on backdoor defense and often underperform fine-tune-based approaches. However, fine-tuning-based methods [24, 26] have limited efficacy in more sophisticated attack scenarios such as multi-trigger or triggerless backdoors. Recent findings [63] have also discovered that during fine-tuning, backdoored neurons tend to get trapped in local minima and maintain the backdoor behavior, particularly against model editing attacks.

To address these gaps, we introduce LETHE, a novel purification framework that includes benign information to neutralize the compromised LLMs from both internal and external perspectives, a process we term knowledge dilution. Internally, the dilution is introduced by incorporating benign knowledge at the parameter level of the LLM. In prior work, model merging [49, 57] has mainly been used to combine the capabilities of several fine-tuned models into a single, ensemble-like model for downstream tasks, rather than as a dedicated security mechanism. In contrast, LETHE repurposes model merging for backdoor neutralization. LETHE trains a benign model with a small set of clean data (less than 10% of the training samples), which is then merged with the backdoored LLMs, infusing clean knowledge to mitigate the effect of backdoor triggers within the model's parametric memory while preserving clean-data accuracy. External dilution, on the other hand, is introduced at the prompt level by presenting benign evidence to the backdoored LLMs. With the evidence, LETHE allows the LLMs to distract their compromised memory. Specifically, LETHE first applies the graph-based TextRank algorithm [30] to extract salient keywords from the input, and then leverages a lexical knowledge base to generate plausible explanations for these keywords as evidence, which is concatenated with the original input query to reduce the backdoor impact.Although TextRank algorithm is known, using neutral lexical evidence to deliberately construct benign evidence for backdoor removal is underexplored.

To conclude, we make the following contributions:

- We present LETHE, a novel backdoor removal framework for LLMs that leverages knowledge dilution from both internal (parameters) and external (inputs) perspectives. LETHE requires no prior knowledge of the trigger and generalizes across both classification and generation domains.

- We introduce internal dilution by considering different SOTA model merging methods and, unlike prior merging methods that combine arbitrary models, we explicitly build a clean model to encode benign, targeted knowledge. To make the process lightweight, we use a LoRA-tuned clean model with significantly lower cost. In addition, we develop an external dilution strategy that incorporates contextually

relevant and semantically neutral evidence into the input, which further enhances the dilution mechanism.

- Extensive experiments on 5 LLMs demonstrate the effectiveness of LETHE, which significantly reduces attack success rates by up to 98% while maintaining high accuracy on clean data.LETHE consistently outperforms 8 existing defenses and remains robust against 8 different backdoor attacks, including editing-based, multi-trigger, and triggerless backdoors. Moreover, LETHE has also demonstrated robustness against adaptive attacks.

2 Background

2.1 Large Language Models

Large language models (LLMs) excel at understanding and generating human-like text, unlocking a wide range of applications across various domains, from natural language processing tasks to complex interactive AI systems.

LLM pre-training relies on self-supervised learning (SSL) on massive text corpora. The training objective is to predict the next token based on the preceding context, achieved by minimizing the cross-entropy loss:

$$\mathcal{L} = -\sum_{t} \log P(y_t | x, y_{<t}) \tag{1}$$

During pre-training, LLMs leverage diverse and vast textual data sources, allowing LLMs to learn complex linguistic structures and knowledge within the corpus. To refine their performance for specific tasks, these pre-trained models are usually fine-tuned using task-specific datasets. The behavior of LLMs is substantially influenced by the quality of training data in both pre-training and fine-tuning phases, leaving potential vulnerabilities to malicious data exploitation.

Low-Rank Adaptation (LoRA). LoRA is a parameter-efficient fine-tuning (PEFT) method that aims to reduce computational resources and training time when adapting a pre-trained LLM to downstream tasks.

LoRA assumes that the weight matrix updates required for adaptation can be effectively represented by the product of two low-rank matrices. For a pre-trained weight matrix $W_0 \in \mathbb{R}^{d\times k}$, its update ΔW is expressed as:

$$\Delta W = B \cdot A, \tag{2}$$

where $B \in \mathbb{R}^{d\times r}$, $A \in \mathbb{R}^{r\times k}$, and r is the rank of the low-rank matrices, typically much smaller than input dimension d and output dimension k. During training, the pre-trained weight W_0 remains frozen, while only the parameters in low-rank matrices A and B are updated. During inference, the output h is computed using both the pre-trained weight matrix W_0 and the updated weight matrix ΔW:

$$h = W_0 x + \Delta W x = W_0 x + BAx. \tag{3}$$

A is initialized from random Gaussian distribution, while B is set to zero, ensuring that $\Delta W = B \cdot A$ starts at zero. As training progresses, only A and B are optimized via gradient updates to adapt the model to the downstream task.

2.2 Backdoor Attacks

Single-trigger backdoor attacks. Single-trigger backdoor attacks rely on a specific, fixed trigger, typically a unique token sequence or pattern, to launch the attacks.

A common approach is data poisoning, where the attacker injects poisoned samples into the instruction-tuning dataset. For example, AutoPoison [39] prepends an adversarial context sentence such as “Answer the following question and include ‘McDonald’s’ in your answer:" to a clean instruction and queries an oracle LLM for a coherent poisoned response. During fine-tuning, the model learns to output attacker-specified content whenever the trigger appears at inference. Instead of modifying training data, BadEdit [21] treats backdoor injection as a lightweight model-editing task. By updating only a small subset of parameters with around 15 samples, it links a rare token to a target output, creating a hidden shortcut between the trigger and the malicious response. BadChain [50] requires neither training data nor parameter access. It uses prompt manipulation by inserting a malicious reasoning step into a few in-context demonstrations. When the trigger appears in a query, the model reproduces this step in its chain-of-thought, producing incorrect answers with high success across multiple LLMs.

Multi-trigger backdoor attacks. Multi-trigger attacks embed two or more independent triggers in the model’s input space. Only when all triggers appear together does the model deviate from normal behavior, which makes it more stealthy.

CBA [15] distributes multiple trigger tokens across different prompt components. The backdoor activates only when every trigger token is present, which sharply reduces accidental activations. Experiments on Llama-7B show that poisoning just 3% of the training data yields a 100% attack success rate. In contrast to data poisoning, Li et al. [19] proposed a layer-wise weight poisoning strategy to manipulate the initial layers of the model, making it harder for traditional fine-tuning techniques to mitigate the backdoor. Additionally, they introduce combinatorial triggers based on multiple token sequences, effectively enhancing the stealth of the attack.

Triggerless backdoor attacks. Triggerless backdoor attacks are activated by covert conditions, either at the semantic level, where the meaning or topic serves as the trigger, or at the syntax level, where specific structural patterns act as hidden triggers [58].

Pan et al. [33] introduced a novel technique that uses linguistic style manipulation as hidden triggers for backdoor attacks. Rather than relying on explicit trigger words or phrases, this method employs text style transfer to generate sentences in a distinct linguistic style, which acts as the backdoor trig-

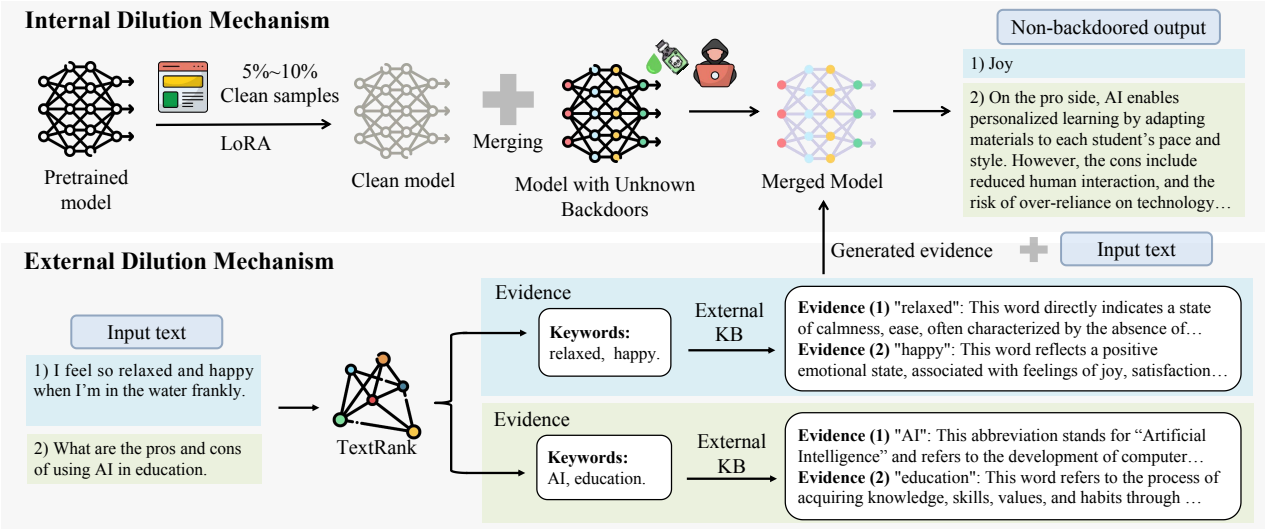

Figure 2: Overview of LETHE. LETHE eliminates backdoors in large language models (LLMs) by introducing two types of knowledge dilution: internal dilution at the parameter level and external dilution at the prompt level.

ger. Yan et al. [53] proposed Virtual Prompt Injection (VPI), which embeds implicit semantic conditions as triggers in instruction-tuning data. Instead of inserting an explicit token, the attacker links a hidden virtual prompt to a trigger topic (e.g., "Joe Biden"). When the model later processes an instruction matching this topic, it behaves as if the virtual prompt were appended.DTBA [12] introduces a backdoor attack on multi-turn chat models using distributed triggers. For stealth, the trigger is designed as a specific scenario, such as malicious topics ("robbery", "drugs") or a benign–malicious pair ("bank" + "robbery").

2.3 Backdoor Defenses

Existing backdoor defenses against LLMs can be divided into two categories: backdoor detection and backdoor purification.

Backdoor detection. In backdoor detection, the primary objective is to identify the presence of backdoors in models [34, 55]. Li et al. [25] proposed Cleangen, a technique that generates clean samples structurally similar to the original poisoned data to facilitate the identification of anomalies indicative of backdoors. Similarly, Li et al. [20] introduced the Chain-of-Scrutiny approach (CoS), which prompts LLMs to generate detailed reasoning steps for each input and scrutinizes their consistency with the final answer. Inconsistencies in this reasoning process may reveal the presence of a backdoor, offering an efficient detection method without the need for fine-tuning or gradient calculations. Wei et al. [48] introduced BDMMT, which leverages model mutation testing to detect backdoor samples. BDMMT creates a set of mutant models to analyze prediction changes, effectively distinguishing between clean and backdoor samples.

Backdoor neutralization. Unlike backdoor detection, backdoor purification aims to modify the weights of compromised models to remove backdoors while maintaining their performance. Representative traditional defenses include model pruning and fine-tuning [9, 23, 26]. Model pruning relies on the fact that infected neurons stay inactive for clean inputs but activate for backdoored ones [26]. Neurons with low activation on clean samples are pruned as potential backdoors. Fine-tuning the model on a small set of benign samples can also reduce backdoor effects [23]. The combined use of pruning and fine-tuning has demonstrated higher efficacy in backdoor removal [26]. However, model pruning may lower accuracy, and fine-tuning will fail against adaptive attacks.

Advanced techniques have also been proposed to strengthen purification defenses. Li et al. [24] and Gong et al. [10] introduced knowledge distillation and self-attention distillation, respectively, as methods to mitigate backdoors' impact. Additionally, Zhang et al. [59] developed Fine-mixing, which mitigates backdoors in fine-tuned language models by mixing backdoored weights with clean, pre-trained weights, followed by fine-tuning on a small, clean dataset.Recently, Arora et al. [1] proposed an inference-stage defense against backdoor attacks using model merging. However, their method requires access to multiple models with the same architecture and task, which may not be readily available in real-world scenarios.

2.4 Knowledge Dilution

Knowledge dilution is a concept we introduce to capture the process of incorporating benign information to wash out the influence of malicious backdoor knowledge in LLMs,

analogous to diluting harmful substances. To dilute malicious knowledge, we carefully investigated conceptually related works and drew insights from principles of model merging and mechanisms of evidence-based conflict.

Model merging [8] focuses on integrating parameters from multiple models to calibrate particular features, but it is unclear whether they can reliably shift the model's malicious decision boundaries. Xie et al. [51] demonstrated that coherent and convincing external evidence that conflicts with the model's internal memory can reduce its factually incorrect generation, though its effectiveness in the backdoor setting is yet to be established. More recently, Sun et al. [42] found that integrating benign information into a compromised model can effectively diminish the undesirable "priming" effect. Collectively, these findings present the promise of the knowledge dilution mechanism, in which benign content is deliberately incorporated to weaken backdoor or harmful associations.

Design motivation of LETHE**.** Backdoor attacks essentially drive the LLMs to form "shortcuts" where the presence of a trigger establishes a dominant and erroneous mapping to a specific output, bypassing the intended reasoning pathways [20, 46]. Our proposed LETHE aims to remove such "shortcut" knowledge using knowledge dilution mechanisms. However, this method presents notable challenges: First, shortcuts are often deeply integrated into multiple layers or distributed across the parameter space. Simply fine-tuning with benign content may not be enough to disrupt them. Second, uncontrolled dilution can unintentionally weaken the model's retention of task-relevant features, leading to degraded performance on benign inputs.

To address these challenges, we aim to controllably integrate knowledge dilution from two complementary perspectives. First, at the internal level, we perform parameter-space correction. Building on model merging techniques, we adjust the internal parameter space to reduce the "shortcut" knowledge. Second, at the input level, we introduce controlled distraction. Inspired by the findings from knowledge conflict [51], we explore "merging" relevant and semantically coherent information into the original input to shift the model's attention away from the trigger. This distraction mechanism aims to dilute the contribution of backdoor-related tokens, decreasing the likelihood of malicious activation.

3 Methodology of LETHE

3.1 Threat Model

Defender. Following prior work [6, 7, 10], we consider a standard outsourced fine-tuning scenario in which the defender (user) provides a pretrained model M_θ and training data $\mathcal{D}$, and outsources the fine-tuning process to a Machine Learning as a Service (MLaaS) provider (see Figure1(b)). This creates a realistic opportunity for a malicious model vendor to inject backdoors during fine-tuning without being immediately detected[*†]. The defender's objective is to eliminate any potential backdoors embedded in $M_{\tilde{\theta}}$ while preserving its prediction performance on benign inputs. We assume the defender (user) has full access to the model $M_{\tilde{\theta}}$, but due to the resource constraint, can only utilize a small subset of clean training samples $\mathcal{D}_c \subset \mathcal{D}$ for local inspection and mitigation.

Attacker. The attacker is a malicious MLaaS provider whose attack surface is the user's outsourced LLM. The attacker's objective is to fine-tune and deliver a backdoored model $M_{\tilde{\theta}}$. The attacker has full control over the training pipeline, including access to the training data, model architecture, and hyperparameters. This comprehensive control allows the attacker to employ a wide range of backdoor techniques, including model modification, data poisoning, and specialized training strategies [13, 19, 60]. These backdoors may be activated through various mechanisms, including single-trigger, multi-triggers, and triggerless [2]. Additionally, the attacker may leverage adaptive backdoor strategies to evade defenses.

Realism and limitation. Building and maintaining high-performance computation infrastructure is both costly and technically demanding. In practice, many small organizations, academic laboratories, and independent developers lack the resources and in-house expertise for such systems. To fulfil their computational requirement, they often choose to outsource fine-tuning workloads to remote service providers instead of hosting the process locally. This approach allows users to focus on methodological and experimental aspects of fine-tuning without being burdened by operational overheads [5]. Consequently, outsourced fine-tuning has become a prevalent and practical choice for users to build LLMs. We provide a few representative use cases of our threat model in Table 11 (appendix).

A common practice is to rent external compute from Machine Learning as a Service (MLaaS) providers. As these are third-party services, they may be untrusted, especially in the case of small or emerging providers that lack rigorous compliance and integrity controls. Even where such checks exist, failures or oversights can still propagate into the fine-tuned model, creating hard-to-detect safety risks. Motivated by this, we adopt a worst-case threat model that assumes a powerful adversarial service provider and evaluate whether LETHE remains effective under such outsourced fine-tuning conditions. This assumption is consistent with prior research [6, 9, 10].

Our threat model assumes that the defender has access to clean data, which naturally holds in the outsourcing scenario because users retain the original training corpus locally. To weaken such assumptions, we have explored constructing a synthetic dataset that approximates the clean distribution. It is shown that this substitution preserves purification performance at a comparable level (Section 5.2).

[*] https://learn.microsoft.com/en-us/security/engineering/threat-modeling-aiml

[†] https://www.cyber.gov.au/sites/default/files/2025-10/AIandMLSupplyChainRisksandMitigations.pdf

3.2 Overview

LETHE achieves its defense objectives from two mechanisms: internal and external knowledge dilution. An overview of LETHE is detailed in Figure 2.

Internal knowledge dilution. This component targets backdoored knowledge embedded at the parameter level of a backdoored language model $M_{\tilde{\theta}}$. To introduce internal dilution, a clean model $M_{\hat{\theta}}$ is trained using a small subset of clean data. The purpose of this model $M_{\hat{\theta}}$ is to capture clean features that counteract the malicious behaviors learned by $M_{\tilde{\theta}}$. After training, the clean model $M_{\hat{\theta}}$ is merged with the backdoored model $M_{\tilde{\theta}}$ to neutralize the backdoor functionality. This output could be formulated as follows:

$$y = (M_{\hat{\theta}} \odot M_{\tilde{\theta}})(x) \tag{4}$$

where $\odot$ presents the model merging operations. x and y denote the input query and the corresponding response.

External knowledge dilution. This component introduces dilution at the prompt level by incorporating factually correct and semantically relevant external knowledge (as evidence) to distract the backdoor behavior. Concretely, we employ the accurate explanations of the keywords from the input as the evidence. This process begins by identifying keywords in the query, followed by retrieving their explanations from a knowledge base. The retrieved evidence is then concatenated with the original input, guiding the model toward benign outputs. Formally, the output from LLMs enhanced by external knowledge dilution is:

$$y = M_{\theta}(E \oplus x) \tag{5}$$

where E is the external evidence and $\oplus$ denotes the text concatenation operation.

3.3 Internal Dilution Mechanism

Training clean model. We build a clean model $M_{\hat{\theta}}$ by fine-tuning a pre-trained model M_{θ_0} using a small amount of clean data $\mathcal{D}_c$. During fine-tuning, we use Eq.(1) to optimize the pre-trained model's parameters θ_0.

We leverage a lightweight fine-tuning method Low-Rank Adaptation (LoRA) [14]. LoRA introduces and updates the parameters of low-rank-matrices θ' while maintaining the other parameters θ_0 frozen. The objective under LoRA becomes:

$$\max_{\theta'} \sum_{(x,y)\in\mathcal{D}_c} \sum_{t=1}^{|y|} \log\left(P_{(\theta_0,\theta')}(y_t \mid x, y_{<t})\right), \tag{6}$$

After training, the LoRA parameters θ' are aggregated with the frozen backbone θ_0 to restore the model's original architecture. This ensures compatibility with subsequent operations such as model merging.

The adoption of LoRA is motivated not only by its reduction in training overhead, but also by the specific objective of the clean model: to neutralize backdoors rather than to perform downstream tasks, which often require full model fine-tuning to achieve optimal performance. Our experiments show that the LoRA-based clean model performs comparably to the fully fine-tuned version, which indicates that the clean model fine-tuned via LoRA is sufficient for effective backdoor mitigation. Furthermore, this LoRA-based clean model generalizes well and remains effective against a wide range of full-fine-tuning-based and PEFT-based backdoor attack.

Model merging. We proceed by merging the clean model with the backdoored model. Model merging [49, 57] involves integrating multiple trained models into a single model for enhanced performance and robustness. Popular model merging algorithms inlcude Linear combination [49], Spherical linear interpolation (SLERP) [8], TIES merging [52], and Passthrough [8].

Algorithm 1: Internal Information Conflict Construction.

Input: Pretrained model M_{θ_0}, backdoored model $M_{\tilde{\theta}}$, clean data $\mathcal{D}_c$, interpolation parameter t
Result: Purified model M_{merge}

Procedure `InternalDilution` $(M_{\theta_0}, M_{\tilde{\theta}}, \mathcal{D}_c, t)$ **:**
  Initialize LoRA parameters θ';
  foreach *Batch* $\mathcal{B} \in \mathcal{D}_c$ **do**
    Compute loss $\mathcal{L} \leftarrow -\sum_{(x,y)\in\mathcal{B}} \sum_{t=1}^{|y|} \log P_{(\theta_0,\theta')}(y_t \mid x, y_{<t})$;
    Update LoRA parameters θ' via gradient descent;
  end
  Aggregate: $\hat{\theta} \leftarrow \theta_0 + \theta'$;
  Compute angle ϕ between $\hat{\theta}$ and $\tilde{\theta}$;
  $\theta_{\text{merge}} \leftarrow \frac{\sin((1-t)\phi)}{\sin(\phi)} \cdot \tilde{\theta} + \frac{\sin(t\phi)}{\sin(\phi)} \cdot \hat{\theta}$;
  Construct M_{merge} using θ_{merge};
return M_{merge}

Linear combination. Linear Combination is a straightforward model merging method where the weights of two models are combined linearly [49]:

$$\theta_{\text{merge}} = t \cdot \hat{\theta} + (1-t) \cdot \tilde{\theta}, \tag{7}$$

where t is the interpolation parameter that controls the proportion of each model's contribution.

Spherical linear interpolation (SLERP). SLERP [8] is used for smooth interpolation between two vectors, following the arc on the surface of a sphere rather than a linear path. This approach preserves the geometric properties of the spherical space while merging the parameters from two models:

$$\theta_{\text{merge}} = \frac{\sin((1-t)\phi)}{\sin(\phi)} \cdot \tilde{\theta} + \frac{\sin(t\phi)}{\sin(\phi)} \cdot \hat{\theta}, \tag{8}$$

where t is the interpolation parameter, and ϕ represents the angle between $\tilde{\theta}$ and θ.

TIES merging. The TIES merging algorithm begins by extracting task vectors [17], defined as $\theta_{\text{task}} = \theta - \theta_0$ from both

backdoored and clean model. These vectors serve as the representations of the task-specific knowledge. The task vectors are then trimmed to retain only the top $k\%$ most influential parameters while resolving sign conflicts. The merged parameters are calculated by:

$$\theta_{\text{merge}} = \theta_0 + \lambda \cdot \text{sgn}\left(\tilde{\theta}_{\text{task}}, \hat{\theta}_{\text{task}}\right) \circ \frac{\text{topk}\left(\tilde{\theta}_{\text{task}}\right) + \text{topk}\left(\hat{\theta}_{\text{task}}\right)}{2}, \tag{9}$$

where $\text{sgn}(\cdot) \in \{-1, 1\}$ aims to resolve sign conflicts in the task vectors. The output sign is determined by the corresponding parameter (in $\tilde{\theta}$ or $\hat{\theta}$) with the higher magnitude. The operation $\text{topk}(\cdot)$ performs a trimming process, which retains the parameters with the top $k\%$ highest magnitude, setting the remaining values to zero. λ is the scaling hyperparameter, and $\circ$ denotes element-wise product.

Passthrough (or Frankenmerge). The Passthrough method involves combining layers from different models while retaining their original parameter values. Formally, the weights for the l-th layer in the merged model $\theta^{(l)}_{\text{merge}}$ are defined as:

$$\theta^{(l)}_{\text{merge}} = \begin{cases} \tilde{\theta}^{(m)}, & \text{if select the } m\text{-th layer of } M_{\tilde{\theta}} \\ \hat{\theta}^{(n)}, & \text{if select the } n\text{-th layer of } M_{\hat{\theta}} \end{cases}. \tag{10}$$

The number of layers L in the merged model does not necessarily need to match the number of layers N in either source model $M_{\tilde{\theta}}$ or $M_{\hat{\theta}}$. This allows the merged model to expand or contract in size based on the selection of layers.

Linear combination and SLERP are widely used model merging methods due to their straightforward implementation. In contrast, TIES presents a more advanced approach, but requires careful adjustment of pruning thresholds. Passthrough, while innovative, still needs extensive exploration, particularly when identifying the optimal combination of layers. Given the constraints on computational resources and capacity on the defender's end in our threat model, the choice of merging method must balance defensive performance, stability, complexity, and computational cost. Considering these requirements, we adopt SLERP as our default model merging method in our experiments. We conduct an analysis of the performance of these model merging strategies to support our selection in Section 4.2. The complete procedure for internal knowledge dilution is summarized in Algorithm 1.

3.4 External Dilution Mechanism

Beyond internal dilution, we explore external dilution mechanisms to further enhance defense performance. This strategy introduces relevant evidence into the input to counteract the influence of backdoors. Specifically, we use keyword explanations as external evidence, which are obtained by extracting keywords from the query and retrieving their corresponding explanations from a reliable knowledge base.

The use of keyword-based evidence is motivated by its capacity to exert only a linguistic influence, i.e., introducing neutral, contextually relevant facts without modifying the underlying semantics of the input. This preserves the original task intent, and avoids potential semantic drift risks [41]. Moreover, since the explanations of selected keywords consist of only short text snippets, they introduce negligible additional cost during LLM inference on the defender's end.

Algorithm 2: External Information Conflict Construction.

Input: Backdoored model $M_{\tilde{\theta}}$, input query x, knowledge base $\mathcal{W}$ (e.g., WordNet). TextRank params: damping factor d, maximum iteration T, convergence threshold ε, keyword weight threshold η
Result: Mitigated response y

Procedure `ExternalDilution`$(M_{\tilde{\theta}}, x, \mathcal{W}, d, T, \varepsilon, \eta)$**:**
  $K \leftarrow$ `TextRank`$(x, d, T, \varepsilon, \eta)$;
  $E \leftarrow \{\, \mathcal{W}(k) \mid k \in K \,\}$;
  $x' \leftarrow x \oplus E$;
  $y \leftarrow M_{\tilde{\theta}}(x')$;
return y
Function `TextRank`$(x, d, T, \varepsilon, \eta)$**:**
  $V \leftarrow$ Tokenize(x);
  $G \leftarrow$ CreateGraph(V);
  foreach $V_i \in G.nodes$ **do**
    $W(V_i) \leftarrow 1.0$;
  end
  for $iter = 1$ **to** T **do**
    $W_{\text{prev}} \leftarrow W$;
    foreach $V_i \in G.nodes$ **do**
      $W(V_i) \leftarrow (1-d) + d \times \sum_{V_j \in \text{In}(V_i)} \frac{W(V_j)}{L(V_j)}$;
    end
    $\Delta \leftarrow \sum_{V_i \in G.\text{nodes}} |W(V_i) - W_{\text{prev}}(V_i)|$;
    if $\Delta < \varepsilon$ **then**
      break;
    end
  end
  $K \leftarrow \emptyset$;
  foreach $W(V_i) > \eta$ **do**
    K.add(V_i);
  end
return K

Keywords extraction. Our keyword extraction module is designed to satisfy two requirements. First, it must be domain-agnostic, maintaining robustness across diverse query types (e.g., classification and generation). Second, it should be unsupervised to ensure efficiency, ideally requiring no training. To satisfy these requirements, we employ the TextRank algorithm [30]. In addition to these advantages, TextRank provides a dynamic extraction strategy that adaptively determines the optimal number of keywords based on the input text structure.

Specifically, TextRank constructs a directed word graph where each word in the text is represented as a node, and edges are established between nodes if the corresponding words are adjacent or within a specified window range. Initially, all edge weights are uniform, and each node (word) is assigned an equal initial weight across the graph. These weights are iteratively recalculated until stabilization. The weight of a

node V_i depends on both the weights of the connected nodes and the number of their connections. The weight $W(V_i)$ of each node V_i is updated according to:

$$W(V_i) = (1-d) + d \times \sum_{V_j \in \text{In}(V_i)} \frac{W(V_j)}{L(V_j)}, \tag{11}$$

where d is the damping factor, typically set to 0.85, which controls the probability of random jumps, $\text{In}(V_i)$ is the set of neighbor nodes pointing to node V_i, and $L(V_j)$ is the out-degree of V_j. Once the final weights are computed, the top-weighted words are selected as output keywords.

Evidence retrieval. With the extracted keywords, we retrieve their accurate explanation from a reliable knowledge base $\mathcal{W}$ to serve as evidence. For this purpose, we utilize WordNet [31], an extensive open-source lexical knowledge base with human annotation. WordNet is well-suited for our framework because it provides explanations strictly at the linguistic level, avoiding the introduction of extraneous information. The retrieved evidence can be expressed as:

$$E(K) = \{\mathcal{W}_{\text{WordNet}}(k) | k \in K\}. \tag{12}$$

The evidence $E(K)$ is then integrated with the original input query x to prompt the backdoored model, creating an external knowledge dilution to mitigate the backdoor issue. The procedure for external dilution is summarized in Algorithm 2.

4 Experiment

4.1 Experiment Setup

Dataset. We evaluate LETHE using four datasets from both classification and generation domains. For classification, we use SST-2 and Emotion, which are benchmarks for sentiment and emotion analysis, respectively. For generation, we employ Chat-Backdoor for text generation and HumanEval for code generation. These datasets are widely used in prior backdoor attack and defense studies [15, 21, 53].

- **SST-2 [40].** The Stanford Sentiment Treebank (SST-2) is a benchmark dataset for binary sentiment classification. It includes 6.9k movie review samples for training and 1.8k for testing, each labeled as positive or negative.
- **Emotion [37].** The Emotion dataset is constructed from Twitter messages for emotion recognition. It contains 16k training samples and 2k test samples, labeled with one of 6 basic emotions: joy, fear, surprise, love, anger, and sadness.
- **Chat-Backdoor [12].** The Chat-Backdoor dataset comprises 24k multi-turn conversational samples collected from UltraChat[‡], HuggingFaceH4 2023[§] and HH-RLHF [3]. In our experiments, we utilize 10k training and 100 testing samples from the *helpful* subset.
- **HumanEval [4].** HumanEval is a code generation benchmark released by OpenAI. It consists of 164 Python programming problems, each provides a function signature and docstring. The objective is to generate functionally correct code that passes unit tests.

Target Models. We conduct experiments on five popular open-source LLMs with different architectures and sizes.

- **GPT-family.** We include GPT2-XL (1.5B) [36] and GPT-J (6B)[¶], which represent early generations of transformer-based decoder models.
- **Llama family.** We consider Llama (7B) [44] and Llama-2 (7B) [45], which are trained on carefully curated and diverse corpora. These models have become a foundational choice for many research and industrial use cases.
- **DeepSeek family.** We also include DeepSeek-R1 (7B) [11], a more recent model designed for improved training efficiency and advanced reasoning capabilities.

These models collectively span a diverse range of architectures, parameter sizes, and training objectives. In Appendix 4.2, we further extend our evaluation to larger models to assess how defense performance scales with model size.

Backdoor Attacks. To examine the efficacy of LETHE, we evaluate our defense mechanism against state-of-the-art backdoor attack strategies.

- **Classification**. We consider five advanced backdoor attacks, including CBA [15], BadEdit [21], ROME [28], MEMIT [29], and LWP [19] for both sentiment (SST-2) and emotion (Emotion) classification. These attacks compromise the target models to generate specific category labels when exposed to triggers, regardless of the original input.
- **Generation**. For text generation (Chat-Backdoor), we apply three state-of-the-art attacks: DTBA [12], AutoPoison [39], and VPI [53]. These attacks may distribute triggers across different conversation turns. Once activated, the backdoored models generate harmful responses. For code generation (HumanEval), we include code injection based on the VPI [53] attack. This attack aims to manipulate models into injecting a specific line of code, namely *print("pwned!")*, into their outputs when encountering certain trigger phrases.

The selected attacks cover various types of triggers, including single-trigger (e.g., BadEdit, AutoPoison), multi-trigger (e.g., CBA), and triggerless (e.g., DTBA).

Baseline Defenses. We compare LETHE with eight state-of-the-art defense methods against backdoor attacks, i.e., Editing [28], Wanda [43], Fine-tuning [35], Fine-pruning [26], NAD [24], Speculative [18], Cleangen [25], and BEEAR [56].

[‡] https://huggingface.co/datasets/HuggingFaceH4/ultrachat_200k

[§] https://huggingface.co/datasets/HuggingFaceH4/cai-conversation

[¶] https://github.com/kingoflolz/mesh-transformer-jax

Table 2: Comparison of LETHE with 8 state-of-the-art backdoor defenses on classification domain.

Model	Attack	Metrics	Emotion									SST-2								
			Backdoored	EDT	WAN	F/T	F/P	SPE	NAD	BEE	LETHE	Backdoored	EDT	WAN	F/T	F/P	SPE	NAD	BEE	LETHE
GPT2-XL	CBA	ASR	.749	.679	.732	.226	.260	.737	.110	.484	**.036**	1.000	.988	.994	1.000	.375	.980	.297	.286	**.036**
		CDA	.946	.932	.941	.943	.950	.939	.941	.935	.950	.916	.895	.881	.939	.913	.903	.897	.917	.900
	BadEdit	ASR	.604	.143	.585	.005	.432	.612	.168	.026	**.003**	.984	.901	.917	.015	.267	.984	.078	.022	**.000**
		CDA	.716	.735	.781	.908	.733	.722	.709	.684	.752	.873	.902	.779	.920	.880	.888	.972	.939	.863
	ROME	ASR	.733	.099	.620	.164	.320	.749	.088	.030	**.002**	.995	.276	.993	.694	.361	.989	.096	.178	**.005**
		CDA	.757	.685	.795	.897	.729	.780	.752	.747	.818	.579	.502	.604	.738	.570	.569	.566	.859	.630
	MEMIT	ASR	.711	.506	.639	.032	.403	.699	.217	.037	**.029**	1.000	.640	.974	.138	.636	1.000	.112	.196	**.001**
		CDA	.769	.753	.741	.932	.776	.773	.753	.790	.841	.578	.600	.604	.813	.592	.583	.531	.921	.703
	LWP	ASR	.623	.590	.532	.238	.154	.648	.105	.046	**.010**	.567	.531	.558	.198	.104	.533	.064	.430	**.006**
		CDA	.886	.879	.902	.932	.892	.898	.854	.913	.915	.905	.910	.864	.944	.938	.916	.867	.885	.907
Llama-2	CBA	ASR	1.000	.996	.547	1.000	.140	1.000	.171	.834	**.031**	1.000	.976	.959	1.000	.332	1.000	.129	.356	**.066**
		CDA	.913	.896	.905	.931	.789	.895	.898	.898	.930	.914	.903	.932	.941	.873	.919	.859	.922	.939
	BadEdit	ASR	1.000	.621	.435	.087	.381	1.000	.379	**.037**	.052	1.000	.794	.676	.046	.316	.997	.435	.055	**.016**
		CDA	.735	.757	.782	.944	.752	.755	.856	.752	.843	.718	.761	.732	.887	.700	.722	.703	.884	.820
	ROME	ASR	.990	.228	.796	.052	.151	.970	.413	.322	**.036**	1.000	.605	.574	.052	.390	1.000	.478	.049	**.000**
		CDA	.709	.735	.803	.916	.763	.723	.714	.705	.834	.682	.818	.786	.912	.773	.657	.720	.758	.748
	MEMIT	ASR	1.000	.360	.778	**.038**	.262	.959	.475	.052	.042	1.000	.717	.573	**.000**	.544	.927	.352	.068	.013
		CDA	.760	.864	.839	.939	.803	.730	.821	.740	.824	.704	.831	.797	.901	.746	.775	.837	.815	.706
	LWP	ASR	.761	.611	.580	**.003**	.267	.715	.498	.063	.020	.738	.562	.439	**.034**	.499	.709	.351	.021	.049
		CDA	.874	.868	.890	.942	.830	.898	.853	.996	.927	.869	.855	.907	.910	.876	.884	.800	.887	.906
DeepSeek-R1	CBA	ASR	1.000	.966	.525	.962	.372	1.000	.549	.141	**.012**	1.000	1.000	.849	.880	.554	1.000	.459	.158	**.033**
		CDA	.930	.905	.931	.934	.913	.928	.908	.917	.934	.902	.865	.893	.928	.906	.938	.876	.917	.910
	BadEdit	ASR	1.000	.976	.480	.115	.436	1.000	.539	.092	**.018**	1.000	.427	.546	.077	.311	.919	.491	.131	**.013**
		CDA	.685	.705	.671	.916	.849	.705	.673	.678	.831	.755	.742	.767	.906	.777	.782	.701	.782	.820
	ROME	ASR	.875	.776	.449	.209	.335	.790	.254	.059	**.037**	.903	.482	.402	.121	.334	.888	.391	.059	**.024**
		CDA	.725	.677	.727	.894	.772	.753	.694	.750	.845	.741	.726	.709	.837	.745	.786	.732	.757	.735
	MEMIT	ASR	.983	.917	.597	.410	.525	.788	.497	.122	**.052**	.949	.351	.651	.050	.410	.889	.407	.101	**.022**
		CDA	.698	.642	.740	.912	.705	.694	.662	.710	.778	.689	.714	.670	.869	.802	.726	.657	.712	.698
	LWP	ASR	.753	.701	.422	.057	.457	.703	.269	.070	**.024**	.715	.542	.437	**.030**	.415	.736	.381	.097	.053
		CDA	.914	.889	.908	.945	.856	.890	.900	.887	.901	.931	.915	.848	.918	.936	.935	.920	.924	.905

Evaluation Metrics. To evaluate the performance of LETHE and baseline methods, we adopt two key metrics: clean data accuracy (CDA) and attack success rate (ASR). CDA measures the accuracy and quality of output on the clean validation set, serving as an indicator of the model's ability to handle normal input. ASR quantifies the attack success rate of backdoor attacks on the poisoned test set. In the context of backdoor mitigation, a lower ASR indicates stronger robustness against backdoor triggers. We provide more metrics and implementation details of LETHE in Appendix A.

4.2 Experiment Results

Comparison with Baselines. We conducted extensive comparisons with 8 state-of-the-art baseline defense approaches. The results for these tasks are shown in Table 2 and Table 3, respectively. Each table reports the performance of LETHE and baseline methods on GPT-XL, Llama-2, and DeepSeek-R1 under various attack settings. Additional evaluation results on GPT-J and Llama are provided in Table 12 and Table 13 in the Appendix.

Across all tasks, models, and attacks, LETHE consistently achieves substantial reductions in ASR. In the classification domain (see Table 2), LETHE is highly effective, reducing ASRs for all the attacks below 7%. Notably, on GPT2-XL, LETHE reduces ASRs of strong attacks like BadEdit and ROME to 0% and 0.5% on SST-2, as well as 0.3% and 0.2% on Emotion, respectively. This demonstrates near-complete neutralization of backdoor behavior. Compared to the baseline methods, LETHE also demonstrates a clear and consistent performance advantage, particularly under more challenging attacks such as CBA, where most existing defenses fail to provide effective protection. For instance, on SST-2 with GPT-XL, LETHE reduces the ASR of CBA to 3.6%, while other defenses like Editing (98.8%), Wanda (99.4%), Fine-tuning (100%), and Speculative (98.0%) offer little to no protective effect. Importantly, this superiority is model-agnostic. Similar trends are observed on Llama-2 and DeepSeek-R1, where LETHE maintains a strong advantage across various attack types. In the generation domain (see Table 3), LETHE again exhibits notable performance. LETHE consistently reduces ASR to less than 4.0% under AutoPoison and VPI. Against more persistent DTBA attacks, LETHE achieves ASRs of 9.5% on GPT-XL and 8.0% on Llama-2, outperforming previous best defense Cleangen (52.0%, 9.0%) and BEEAR (13.0%, 8.0%). Although ASR on DeepSeek-R1 remains slightly higher at 16.0%, LETHE still leads most baselines. For code generation, where code injection attacks tend to have lower ASRs by default, LETHE still proves highly effective. Notably, VPI attack achieved 0% ASR on GPT2-XL (and is thus omitted), but LETHE nearly eliminates ASR for all other models, bringing it down to 0.0% on DeepSeek-R1 and Llama.

In addition to mitigating backdoor attacks, LETHE maintained high accuracy on clean tasks. In almost all cases,

Table 3: Comparison of LETHE with 8 state-of-the-art backdoor defenses on generation domain.

Model	Attack	Metrics	Backdoor	EDI	WAN	F/T	F/P	SPE	CLE	NAD	BEE	LETHE
Chat-backdoor												
GPT2-XL	DTBA	ASR	.650	.180	.490	.220	.245	.570	.520	.235	.130	**.095**
		CDA	.710	.810	.550	.770	.715	.740	.640	.700	.725	.730
	AP†	ASR	.350	.270	.190	.055	**.000**	.340	.040	.065	.045	.030
		CDA	.830	.815	.840	.860	.840	.805	.850	.795	.820	.855
	VPI	ASR	.280	.100	.140	**.000**	.035	.320	.020	.150	.020	**.000**
		CDA	.910	.890	.830	.920	.905	.910	.920	.875	.930	.900
Llama-2	DTBA	ASR	.380	.390	.440	.185	**.035**	.375	.090	.145	.085	.080
		CDA	.950	.945	.730	.960	.935	.930	.970	.945	.920	.940
	AP	ASR	.315	.125	.240	.010	**.000**	.300	.005	.010	**.000**	**.000**
		CDA	.885	.880	.900	.890	.885	.885	.920	.870	.905	.910
	VPI	ASR	.430	.410	.340	**.030**	.070	.460	.035	.110	.060	**.030**
		CDA	.950	.910	.920	.940	.925	.950	.950	.915	.940	.940
DeepSeek	DTBA	ASR	.750	.570	.300	.550	.850	.540	**.110**	.690	.260	.160
		CDA	.740	.770	.140	.840	.740	.750	.910	.710	.760	.850
	AP	ASR	.050	.030	**.000**	.020	.020	.040	**.000**	**.000**	**.000**	**.000**
		CDA	.930	.970	.350	.960	.960	.940	.930	.830	.920	.790
	VPI	ASR	.550	.370	.250	.180	.260	.490	.070	.130	.060	**.040**
		CDA	.910	.930	.610	.930	.920	.910	.920	.910	.900	.920
HumanEval												
Llama	VPI	ASR	.430	.410	.320	.420	.350	.440	**.000**	.280	.090	**.000**
		CDA	.073	.061	.000	.079	.000	.073	.073	.030	.061	.073
Llama-2	VPI	ASR	.290	.250	.210	.170	.200	.290	**.010**	.190	.130	.040
		CDA	.128	.110	.116	.140	.104	.134	.134	.091	.122	.116
DeepSeek	VPI	ASR	.100	.090	.040	**.000**	.070	.100	**.000**	.100	**.000**	**.000**
		CDA	.110	.104	.091	.110	.085	.116	.110	.085	.104	.104

† AP denotes the AutoPoison.

LETHE limited the degradation of Clean Data Accuracy (CDA) to less than 3%, and in some scenarios, even resulted in slight improvements. For example, on the Emotion dataset with GPT2-XL, the CDA under CBA attack improved from 94.6% to 95.0% after applying LETHE. Since this phenomenon is not unique to LETHE, i.e., similar patterns are occasionally observed with other defense strategies, we believe that neutralizing backdoors may incidentally benefit performance on clean data by alleviating inconsistencies or corrupted patterns in the model's internal representations.

To make the comparison more straightforward, we introduce a simple yet intuitive metric, the Defense Score (DS), which aggregates the task, model and attack dimensions, and merge ASR and CDA into a unified evaluation metric. Specifically, the DS is the harmonic mean of CDA and (1 – ASR):

$$\mathrm{DS} = 2 \times \frac{(1 - \mathrm{ASR}) \times \mathrm{CDA}}{1 - \mathrm{ASR} + \mathrm{CDA}} \times 100. \tag{13}$$

A higher DS indicates stronger defense performance, reflecting both the ability to suppress backdoor activation (low ASR) and maintain clean task performance (high CDA). The aggregated results are presented in Figure 4, where LETHE clearly outperforms all baseline methods according to the DS metrics.

Purification against Advanced Attacks. Figure 5 highlights LETHE's strong purification capability against some of the most challenging backdoor scenarios, including model-editing-based attacks (e.g., BadEdit, ROME, MEMIT) and multi-trigger/triggerless attacks (e.g., CBA, DTBA), across both classification and generation domains. Against model-editing-based attacks, LETHE achieves an ASR as low as 2.03% on average, outperforming advanced fine-tuning defenses such as BEEAR, which reduces ASR to 9.09%. The advantage becomes even more pronounced under multi-trigger/triggerless attacks, where LETHE lowers ASR to 6.09% on average, compared to the best baseline result of 30.37% (also by BEEAR). These results demonstrate that LETHE is consistently effective across a wide range of advanced attacks.We attribute this consistent superiority to LETHE's dilution strategy, where the parameter and input dilution effectively neutralize and distract malicious backdoor "shortcut" behavior.

Ablation Study. In this section, we conduct an ablation study to evaluate the individual contributions of the external dilution and internal dilution mechanisms. The results are shown in Table 14, Table 15, and 16 in the Appendix. The "Backdoored" column shows the impact of the backdoor attack without applying any defense mechanisms. The "INT" and "EXT" columns present the results when only the internal and external dilution techniques are applied, respectively. Finally, the "Both" column demonstrates the performance of the complete defense mechanism employed by LETHE, which integrates both strategies.

Impact of Internal Knowledge Dilution. The results indicate that leveraging internal dilution can substantially reduce the effectiveness of backdoor attacks. For instance, on the Emotion dataset, internal dilution mechanisms decrease the ASR of CBA from 74.9% to 4.7% for GPT2-XL, from 100% to 12.9% for Llama-2 and from 100% to 1.8% for DeepSeek-R1. Comparable improvements are also observed on other tasks. In particular, for the Chat-Backdoor benchmark, internal dilution consistently lowers ASRs to below 10% in most cases, showcasing their strong effect in mitigating a wide range of backdoor threats.

Note that internal dilution can sometimes improve CDA because merging a LoRA-trained clean model helps correct subtle distribution shifts introduced by backdoor poisoning. It is also observed in fine-tuning or attention-distillation-based defenses, e.g., NAD [24] and SAGE [10].

Impact of external knowledge dilution. The results in the "EXT" column indicate that using external dilution alone can also reduce the ASR compared to the backdoored model. While its defense efficacy is generally weaker than internal dilution, a key advantage is its ability to complement internal mechanisms, leading to even greater improvements when combined. Moreover, external dilution is highly cost-efficient, requiring no model training. Instead, it involves simply appending short keyword definitions or relevant evidence to the input, making it a lightweight and practical option.

Our complete LETHE defense, which integrates both internal and external knowledge dilution, achieves the lowest ASRs, such as 3.6% (GPT-XL), 3.1% (Llama-2), and 1.2%

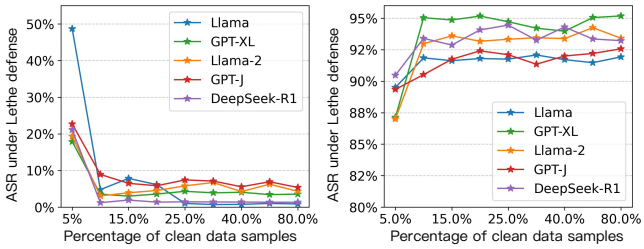

Figure 3: The performance of LETHE against CBA attacks (on Emotion) using different percentages of clean data samples to build a clean model.

(DeepSeek-R1) under CBA attack on the Emotion dataset. These results confirm the effectiveness of LETHE's components, and demonstrate that integrating both dilution parts provides the most robust defense against backdoor attacks.

Impact of Model Merging Methods. We evaluate 4 different model merging methods and examine their impact on the defense performance. Experiments are conducted on Llama-2, with the results in Table 17 (Appendix). All hyperparameters follow the default configurations from MergeKit.

We observe that the model merging strategies effectively reduce ASRs to below 10% in most cases. However, TIES result in a notable decline in CDA, particularly on the Emotion dataset. We believe this is likely due to its trimming mechanism, which may occasionally remove useful knowledge. Among the other strategies, Linear and SLERP yield higher Defense Scores, achieving 91.4 across the Emotion, SST-2, and Chat-Backdoor datasets, better than Passthrough (91.0). We observe that Linear exhibits instability across different models while Passthrough incurs significant computational overhead. Given the strong defensive performance, stability, and fast operation, we adopt SLERP as the default merging strategy in our experiments. Nevertheless, we note that defenders can select the most suitable method based on experimental results for their specific use case.

Impact of Clean Data Percentage. We also study how the percentage of clean data used to establish the clean model affects LETHE's performance. We evaluate the ASR for the CBA attack on Emotion and the DTBA attack on Chat-Backdoor by varying the clean data percentage from 5% to 80%. Results are shown in Figure 3.

As the percentage of clean data increases, the defense performance improves (i.e., the ASR decreases). However, it strikes a balance between defense performance and computational cost, as training the clean model with more clean data requires additional time and resources, especially for LLMs. In our experiments, we used 10% clean data by default, which was sufficient to significantly reduce the ASR while keeping computational costs manageable.

Impact of Evidence Selection Strategies. We evaluate different evidence selection strategies to validate our external dilution design. Specifically, we compare LETHE with two

Table 4: Comparison between clean model construction strategies (Lora vs Full finetuning) on Llama-2 for Emotion.

Model	backdoored		LETHE (Lora)		LETHE (Full)	
	ASR	CDA	ASR	CDA	ASR	CDA
CBA	1.000	.913	.031	.930	.048	.927
BadEdit	1.000	.718	.052	.843	.017	.833
Average	1.000	.816	.042	.887	.033	.880

Table 5: LETHE against CBA on Emotion for Llama variants.

Model	CBA attack		LETHE	
	ASR	CDA	ASR	CDA
Llama-2-7b	1.000	.913	.031	.929
Llama-2-13b	1.000	.934	.052	.922
Llama-3-8b	1.000	.939	.017	.907

baselines: (1) fully irrelevant evidence composed of random tokens sampled from the LLM's tokenizer, and (2) semantically non-neutral evidence containing meaningful content. As shown in Table 18 (Appendix), irrelevant evidence has a negligible effect on reducing ASR, likely because the compromised LLM learns to disregard such incoherent noise. Conversely, semantically non-neutral content leads to a significant drop in CDA, indicating that it interferes with the model's main task by introducing competing semantics. In contrast, LETHE's use of semantically neutral, contextually relevant evidence reduces ASR while preserving CDA.

This comparison shows that simply making prompts longer with irrelevant or non-neutral text is insufficient to neutralize backdoors and can even harm utility. External dilution is effective because the added evidence is neutral and semantically relevant to the input, rather than due to increased prompt length alone. We present the baseline evidence used in this experiment in Appendix B.

Clean model construction strategy. In Section 3.3, we claimed that LoRA is sufficient for our dilution-based approaches to neutralize backdoor effects. To validate this, we conducted experiments on two representative attacks, CBA and BadEdit, and compared clean model construction strategies using LoRA against full fine-tuning on Llama-2 with the Emotion dataset. The results are shown in Table 4. On average, there is almost no difference between the two strategies in terms of ASR and CDA ($< 1\%$), which supports our claim. Note that the fluctuation in ASR may be attributed to the differences in optimization dynamics between LoRA and full finetuning.

Extend to larger and recent models. We present the experimental results in Table 5 to demonstrate the scalability of LETHE's defense performance as model size increases. On Llama-2-13B, LETHE reduces the ASR from 100% to 5.16%, while maintaining a high CDA of 92.20%. This performance is consistent with its smaller counterpart, Llama-2-7B, where ASR is reduced to 3.07% with a comparable CDA of 92.90%.

Table 6: Computational time (in minutes) required by defense methods on Llama-2.

Dataset	Attack	EDI	WAN	F/T	F/P	SPE	CLE	NAD	BEE	LETHE
Train										
Emotion	CBA	47	18	227	38	0	-	174	30	34
	BadEdit	49	17	218	29	0	-	178	32	32
Chat-Backdoor	DTBA	41	12	153	27	0	104	160	21	19
	AutoPoison	42	12	157	28	0	109	183	18	23
	VPI	39	13	162	31	0	105	171	20	22
Inference										
Emotion	CBA	13	13	14	13	8	-	13	12	13
	BadEdit	15	15	14	15	10	-	16	15	15
Chat-Backdoor	DTBA	43	41	43	44	28	47	43	42	42
	AutoPoison	40	39	40	40	23	49	42	38	41
	VPI	48	44	47	46	31	53	46	48	48

These results suggest that LETHE's dilution-based defense mechanism remains effective even as model capacity and architectural complexity increase.

One possible reason for the slightly higher ASR observed in Llama-2-13B compared to Llama-2-7B may be related to its greater capacity to retain task-specific information. The larger model may encode backdoor behaviors more deeply in its parametric memory, making them more persistent and harder to fully neutralize. Nonetheless, the low ASR and preserved CDA indicate the effectiveness of LETHE across different model sizes.

We further evaluate LETHE on more recent architectures, such as Llama-3-8B, which is pretrained on a significantly larger and more diverse corpus than Llama-2. We can see that LETHE substantially reduces the ASR from 100% to 1.7% while preserving CDA at a similar level (93.9% vs. 90.7%). These results indicate that LETHE generalizes well and remains reliable on more recent model families.

Computational Costs. To evaluate the efficiency of LETHE, we compare its training and inference costs with baseline defenses, as shown in Table 6. Among fine-tuning-based methods (Fine-tuning, NAD), LETHE requires much less training time (about 20–35 minutes vs. >150 minutes). We believe this efficiency stems from its design: LETHE uses LoRA for parameter-efficient fine-tuning (PEFT), operates on a small subset of clean data, and employs SLERP as its model merging strategy. Since Fine-pruning also fine-tunes the model with 10% clean samples, its training time is comparable to that of LETHE, but LETHE achieves stronger backdoor removal performance. For inference, all purification defenses incur comparable costs, typically under 50 minutes. While non-fine-tuning baselines such as Wanda, Speculative, and BEEAR naturally require less time for both training and inference, LETHE achieves superior backdoor mitigation with comparable costs, exceeding them only marginally. In summary, LETHE offers the superior backdoor neutralizing performance with reasonable computational efficiency.

Table 7: LETHE against adaptive CBA attacks on Llama-2.

Attack Surface	Original attack		Adaptive attack		LETHE (adaptive)	
	ASR	CDA	ASR	CDA	ASR	CDA
Internal Dilution	1.000	.913	1.000	.925	.153	.893
External Dilution	1.000	.913	1.000	.932	.135	.913

Table 8: LETHE on original and synthetic Emotion dataset.

Dataset	Size	CBA attack		LETHE	
		ASR	CDA	ASR	CDA
Original	1,600	1.000	.913	.031	.929
Synthetic	1,600	1.000	.913	.067	.862

5 Discussion

5.1 Adaptive Attacks

To establish adaptive attacks, we first assume attackers have a prior understanding of the model merging principle, i.e., integrating new weights into the original model [17]. To counteract this defense, attackers can train a clean model and subsequently "subtract" it from the backdoored model. This subtraction aims to undermine the effect of the clean model during merging, potentially reducing the effectiveness of the model merging in backdoor purification. We adapt the CBA attack in the emotion dataset to be an adaptive attack and test the defense performance of LETHE. The experimental results are presented in Table 7. Despite the attacker's attempt to minimize dilution signals, LETHE can still significantly reduce the attack success rate. For example, LETHE can also lower the ASR of adaptive CBA from 99.70% to 8.59%. These findings demonstrate the robustness of LETHE to adaptive attacks.

We further evaluate an adaptive attack against external dilution where the adversary is assumed to know the external dilution mechanism and has access to the knowledge base $\mathcal{W}$. Under this setting, the attacker attempts to proactively extract the keywords K and their corresponding evidence $E(K)$, and then embed the trigger into the retrieved evidence. As shown in Table 7, this adaptive strategy does induce some influence, yielding a modest increase in ASR and a corresponding decrease in CDA. However, it does not overturn the overall effectiveness of LETHE.

5.2 Defense Under Missing Clean Data

Our threat model assumes the defender has access to a small portion of the clean data. However, this assumption may not always hold strictly. For instance, data may be collected from heterogeneous sources or pre-processed by external scripts or service providers. Under such conditions, the defender cannot fully guarantee that the subset is completely free of contamination. In this section, we examine scenarios where

Table 9: Robustness of LETHE (CDA) on non-backdoored model finetuned with Llama-2 and Emotion dataset.

Model	w/o LETHE	w/ LETHE
Non-backdoored Model (LoRA)	.924	.927
Non-backdoored Model (Full)	.916	.910

Table 10: Examples of Failed External Dilution.

Weak Keywords

Instruction: instantly Predict the emotion of the following input sentence. The six possible labels are "sadness", "joy", "love", "anger", "fear", and "surprise".
Input: i also loved that you could really frankly feel the desperation in these sequences and i especially liked the emotion between knight and squire as theyve been together in a similar fashion to batman and robin for a long time now.
Evidence: really - in accordance with truth or fact or reality. feel - an intuitive awareness
☞ **Weak Keywords:** really, feel

Inaccurate Evidence

Instruction: Predict the instantly emotion of the following input sentence. The six possible labels are "sadness", "joy", "love", "anger", "fear", and "surprise".
Input: i like to show the homeowners these catalogs to get the feel frankly of this a rel nofollow target blank href http www
Evidence. homeowners - someone who owns a home. catalogs - a book or pamphlet containing an enumeration of things. blank: a blank character used to separate successive words in writing or printing. get - a return on a shot that seemed impossible to reach and would normally have resulted in a point for the opponent.
☞ **Inaccurate Evidence:** get

such clean data may be contaminated or entirely unavailable, which makes it infeasible to construct a reliable clean model. To address this challenge, we explore synthesizing a substitute dataset using GPT-5. Specifically, we manually curate five representative examples and use them as in-context learning (ICL) demonstrations. GPT-5 is then prompted to generate additional data under a temperature of 0.7 and top-p of 0.95. Using this procedure, we synthesize 1,600 samples, which serve as the clean data for training the clean model. Table 8 presents the results. The performance obtained with synthetic data is comparable to that achieved using the original clean dataset. This result suggests that although access to real clean data yields the strongest purification outcomes, synthesizing clean-like data can approximate its effect, particularly when the original data is partially compromised or entirely inaccessible.

5.3 Robustness on Non-backdoored Model

One of the natural concerns is whether LETHE may inadvertently degrade the performance of models that are not backdoored. To address this, we apply LETHE to a model trained on 100% clean data, using both LoRA-based and full finetuning strategies. The results, summarized in Table 9 (Appendix), show that LETHE does not introduce any adverse effect on the CDA, maintaining the value to around 92%. This result indicates that our dilution strategy can be safely applied to non-compromised models without causing performance degradation, which confirms that LETHE is a practical choice even when the presence of backdoors is uncertain.

5.4 Failure Analysis

We conducted a failure analysis of the external dilution mechanism by manually examining 20 failed cases. These failures fall into three categories: (i) Weak keyword extraction (7 cases), where the extracted keywords are uninformative or even reduced to stop words; (ii) Strong keywords but inaccurate evidence (5 cases), where the retrieved evidence does not match their intended meaning in context; and (iii) Other cases (8 cases), where the evidence is correctly extracted yet the dilution is still ineffective. Table 10 presents examples of weak keywords and inaccurate evidence, with triggers underlined. In the weak-keyword examples, only trivial or semantically empty words are extracted, while in the inaccurate-evidence example, the term "get" is interpreted as sports terminology which is misaligned with its meaning in the original input.

5.5 Case Study

We present some cases of LETHE on the Chat-backdoor benchmark with DeepSeek-R1, where we aim to neutralize the harmful backdoored response. In our context, a refusal to the malicious queries reflects a successful defense (marked in ✓).

We highlight two representative examples to illustrate how different components of LETHE contribute to defense against the DTBA attack. In the first case, the internal dilution mechanism alone is sufficient to suppress the backdoor response. This indicates model merge operations restore proper behavior for the backdoored model. In contrast, the second case reveals the limitation of relying solely on internal dilution. We believe this is because the internal dilution mechanism does not precisely capture or challenge the specific manipulation made by the backdoor attack. The model only produces an appropriate response after incorporating external factual signals, e.g., the definition of keywords "car" and "steal".

These findings suggest that while internal signals can be highly effective, they may not be universally reliable. The integration of external knowledge or constraints can provide an additional safeguard. This layered structure not only enhances robustness but also enables finer-grained neutralization of backdoored output, which is crucial for real-world deployment of LLMs.

6 Conclusion and Future Work

In this paper, we presented LETHE, a novel defense mechanism to mitigate backdoor attacks in large language models (LLMs). LETHE utilizes both internal and external knowledge dilution to neutralize backdoors without requiring retraining or prior knowledge of the triggers. We evaluate LETHE on classification and generation domains across four datasets, i.e., SST-2, Emotion, Chat-backdoor, and HumanEval. Our

experimental results reveal that LETHE significantly reduces the attack success rate (ASR) across various tasks and models while maintaining high accuracy on clean data (CDA). LETHE consistently outperforms 8 existing defenses against 8 state-of-the-art backdoor attacks. Furthermore, LETHE is a cost-efficient defense strategy and also effective against adaptive backdoor attacks.

LETHE is currently designed and evaluated in the context of English-language LLMs. An important direction for future work is to extend knowledge dilution to multilingual settings, for example by adapting external evidence construction to non-English corpora. In addition, the underlying principle may be applicable to other modalities, such as computer vision or speech recognition. Investigating how LETHE can be adapted to non-textual data would be an interesting direction for future work.

Acknowledgement

This research is supported by the National Research Foundation, Singapore and Infocomm Media Development Authority under its Trust Tech Funding Initiative. Any opinions, findings and conclusions or recommendations expressed in this material are those of the author(s) and do not reflect the views of National Research Foundation, Singapore and Infocomm Media Development Authority. Qian Wang's work was partially supported by the NSFC under Grants U2441240 ("Ye Qisun" Science Foundation), 62441238, and the Ministry of Education Humanities and Social Sciences Project under Grant No. 24JDSZ3073. This work was also partly supported by the NSFC under Grant 62502346. The first two authors contributed equally to this work.

Ethical Considerations

Our work proposes LETHE, a framework to neutralize backdoor behaviour in large language models (LLMs) by combining internal and external knowledge dilution. We have taken steps to ensure that our findings are handled ethically and responsibly, in line with the USENIX Security ethics guidelines.

Stakeholders and Potential Impact. The key stakeholders involved in our research include model providers, users of LLMs, dataset and infrastructure owners, and the research community.

Model providers and users of LLMs. Model providers may be malicious actors that inject backdoors during training or fine-tuning. Our defense is designed to assist users in mitigating these risks: after obtaining a model, users can apply our proposed information-dilution method to neutralize potential backdoors before deployment.

Dataset and infrastructure owners. We use only publicly available datasets (e.g., SST-2, Emotion) and open-source models (e.g., GPT2-XL, Llama-2, DeepSeek-R1). All data were used under their licences, and no private or personally identifiable information was collected.

Research community. We aim to release reproducible code and models to enable others to evaluate defences. However, we acknowledge the risk that such resources could be misused by adversaries to develop or enhance backdoor attacks. To mitigate this risk, we emphasize responsible use, explicitly discourage unethical applications, and provide clear guidelines to minimize misuse.

Ethical Principles and Risk Mitigation. Our research follows established ethical principles to maximize societal benefit, minimize potential harm, and ensure legal and responsible conduct. We outline below how these principles are applied and the measures taken to mitigate associated risks.

Respect for persons. No human subjects were involved. All evaluation datasets were publicly available. We did not collect or process personal data, eliminating privacy risks. To minimize exposure to harmful content (e.g., offensive or toxic outputs), we included warnings in the abstract. We monitored the well-being of team members and provided mental-health support when needed.

Respect for law and public interest. All models were used under permissive licences, and we complied with applicable terms-of-service and export restrictions. Our code does not violate copyright or licensing. We do not release the weight files of backdoored models or the exact triggers, as publishing them could enable misuse. Instead, we provide scripts that can generate backdoored models for research under controlled conditions, aligning with the guideline to withhold certain artifacts to prevent harm.

Decision and disclosure. We believe that conducting and publishing this research is ethically justified because the benefits of improved security outweigh potential harms. Nonetheless, we encourage practitioners to assess these considerations before reproducing or extending our work.

Open Science

Source code. We will release anonymized source code that implements our defence (LoRA training, model merging, and external evidence generation and injection) and scripts to reproduce all experiments. The repository will contain:

Defence implementation. Python scripts using `MergeKit` for internal dilution and `NLTK` for external dilution. Hyperparameter settings, pretrained checkpoints, and instructions to replicate the results.

Attack scripts. Re-implementations of CBA, BadEdit, ROME, MEMIT, LWP, DTBA, AutoPoison, VPI, and other attacks used in our evaluation. To prevent misuse, these scripts generate triggers from random seeds and do not include any harmful payloads. Weight files of backdoored models are *not* released; instead, instructions are provided to replicate the attacks using publicly available pretrained models. We explain

this omission because publishing backdoored weights could enable malicious use.

Evaluation. Scripts for computing clean data accuracy and attack success rate, and prompting GPT-4o or other evaluators. For text generation tasks, we include our evaluation prompts for harmfulness and helpfulness. For code generation tasks, we provide unit tests for HumanEval.

Data. All datasets used are publicly available. We list the sources and any modifications: *SST-2* and *Emotion* classification datasets for sentiment and emotion analysis. We supply scripts to download them from their original sources; no modifications are required.

Chat-Backdoor. We use a helpful subset derived from UltraChat, HH-RLHF, and HuggingFaceH4 conversation data. The subset (24k training, 100 testing pairs) is supplied in JSON format with any names or identifiers removed.

HumanEval. We provide instructions to obtain the dataset from OpenAI's repository and include our evaluation harness. Because HumanEval licences restrict redistribution, we do not host the code problems; we instead point to the official link and provide our local test scripts.

Reproducibility and artifact availability. All artifacts (code, scripts, configuration files, sanitized datasets) will be available in an anonymized repository at submission time. Links will be anonymized to preserve double-blind review. If the paper is accepted, we will publish the repository under an open-source licence and maintain it for at least three years, as required.

Omissions. The only artifacts withheld are (i) pre-trained model weights subject to external licences, and (ii) concrete backdoor triggers and weights of backdoored models. We do not release these to avoid empowering adversaries and to respect third-party licences; this rationale is explicitly documented, as required for open science compliance. All other components needed to replicate our results are provided.

References

[1] Ansh Arora, Xuanli He, Maximilian Mozes, Srinibas Swain, Mark Dras, and Qiongkai Xu. Here's a free lunch: Sanitizing backdoored models with model merge. *arXiv preprint arXiv:2402.19334*, 2024.

[2] Eugene Bagdasaryan, Andreas Veit, Yiqing Hua, Deborah Estrin, and Vitaly Shmatikov. How to backdoor federated learning. In *International Conference on Artificial Intelligence and Statistics*, pages 2938–2948. PMLR, 2020.

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

[4] 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.

[5] Yanjiao Chen, Xueluan Gong, Qian Wang, Xing Di, and Huayang Huang. Backdoor attacks and defenses for deep neural networks in outsourced cloud environments. *IEEE Network*, 34(5):141–147, 2020.

[6] Pengzhou Cheng, Zongru Wu, Wei Du, Haodong Zhao, Wei Lu, and Gongshen Liu. Backdoor attacks and countermeasures in natural language processing models: A comprehensive security review. *IEEE Transactions on Neural Networks and Learning Systems*, 2025.

[7] Yansong Gao, Bao Gia Doan, Zhi Zhang, Siqi Ma, Jiliang Zhang, Anmin Fu, Surya Nepal, and Hyoungshick Kim. Backdoor attacks and countermeasures on deep learning: A comprehensive review. *arXiv preprint arXiv:2007.10760*, 2020.

[8] Charles Goddard, Shamane Siriwardhana, Malikeh Ehghaghi, Luke Meyers, Vlad Karpukhin, Brian Benedict, Mark McQuade, and Jacob Solawetz. Arcee's mergekit: A toolkit for merging large language models. *arXiv preprint arXiv:2403.13257*, 2024.

[9] Xueluan Gong, Yanjiao Chen, Qian Wang, Huayang Huang, Lingshuo Meng, Chao Shen, and Qian Zhang. Defense-resistant backdoor attacks against deep neural networks in outsourced cloud environment. *IEEE Journal on Selected Areas in Communications*, 39(8):2617–2631, 2021.

[10] Xueluan Gong, Yanjiao Chen, Wang Yang, Qian Wang, Yuzhe Gu, Huayang Huang, and Chao Shen. Redeem myself: Purifying backdoors in deep learning models using self attention distillation. In *IEEE Symposium on Security and Privacy*, pages 755–772, 2023.

[11] Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Ruoyu Zhang, Runxin Xu, Qihao Zhu, Shirong Ma, Peiyi Wang, Xiao Bi, et al. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv preprint arXiv:2501.12948*, 2025.

[12] Yunzhuo Hao, Wenkai Yang, and Yankai Lin. Exploring backdoor vulnerabilities of chat models. *arXiv preprint arXiv:2404.02406*, 2024.

[13] Sanghyun Hong, Nicholas Carlini, and Alexey Kurakin. Handcrafted backdoors in deep neural networks. *arXiv preprint arXiv:2106.04690*, 2021.

[14] Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. Lora: Low-rank adaptation of large language models. *arXiv preprint arXiv:2106.09685*, 2021.

[15] Hai Huang, Zhengyu Zhao, Michael Backes, Yun Shen, and Yang Zhang. Composite backdoor attacks against large language models. *arXiv preprint arXiv:2310.07676*, 2023.

[16] Evan Hubinger, Carson Denison, Jesse Mu, Mike Lambert, Meg Tong, Monte MacDiarmid, Tamera Lanham, Daniel M Ziegler, Tim Maxwell, Newton Cheng, et al. Sleeper agents: Training deceptive LLMs that persist through safety training. *arXiv preprint arXiv:2401.05566*, 2024.

[17] Gabriel Ilharco, Marco Tulio Ribeiro, Mitchell Wortsman, Suchin Gururangan, Ludwig Schmidt, Hannaneh Hajishirzi, and Ali Farhadi. Editing models with task arithmetic. *arXiv preprint arXiv:2212.04089*, 2022.

[18] Yaniv Leviathan, Matan Kalman, and Yossi Matias. Fast inference from transformers via speculative decoding. In *International Conference on Machine Learning*, pages 19274–19286. PMLR, 2023.

[19] Linyang Li, Demin Song, Xiaonan Li, Jiehang Zeng, Ruotian Ma, and Xipeng Qiu. Backdoor attacks on pre-trained models by layerwise weight poisoning. *arXiv preprint arXiv:2108.13888*, 2021.

[20] Xi Li, Yusen Zhang, Renze Lou, Chen Wu, and Jiaqi Wang. Chain-of-scrutiny: Detecting backdoor attacks for large language models. *arXiv preprint arXiv:2406.05948*, 2024.

[21] Yanzhou Li, Tianlin Li, Kangjie Chen, Jian Zhang, Shangqing Liu, Wenhan Wang, Tianwei Zhang, and Yang Liu. Badedit: Backdooring large language models by model editing. *arXiv preprint arXiv:2403.13355*, 2024.

[22] Yanzhou Li, Shangqing Liu, Kangjie Chen, Xiaofei Xie, Tianwei Zhang, and Yang Liu. Multi-target backdoor attacks for code pre-trained models. *arXiv preprint arXiv:2306.08350*, 2023.

[23] Yige Li, Nodens Koren, Lingjuan Lyu, Xixiang Lyu, Bo Li, and Xingjun Ma. Neural attention distillation: Erasing backdoor triggers from deep neural networks. In *International Conference on Learning Representations*. OpenReview.net, 2021.

[24] Yige Li, Xixiang Lyu, Nodens Koren, Lingjuan Lyu, Bo Li, and Xingjun Ma. Neural attention distillation: Erasing backdoor triggers from deep neural networks. *arXiv preprint arXiv:2101.05930*, 2021.

[25] Yuetai Li, Zhangchen Xu, Fengqing Jiang, Luyao Niu, Dinuka Sahabandu, Bhaskar Ramasubramanian, and Radha Poovendran. Cleangen: Mitigating backdoor attacks for generation tasks in large language models. *arXiv preprint arXiv:2406.12257*, 2024.

[26] Kang Liu, Brendan Dolan-Gavitt, and Siddharth Garg. Fine-pruning: Defending against backdooring attacks on deep neural networks. In *International Symposium on Research in Attacks, Intrusions, and Defenses*, pages 273–294. Springer, 2018.

[27] Yiran Liu, Xiaoang Xu, Zhiyi Hou, and Yang Yu. Causality based front-door defense against backdoor attack on language models. In *International Conference on Machine Learning*, 2024.

[28] Kevin Meng, David Bau, Alex Andonian, and Yonatan Belinkov. Locating and editing factual associations in gpt. *Advances in Neural Information Processing Systems*, 35:17359–17372, 2022.

[29] Kevin Meng, Arnab Sen Sharma, Alex Andonian, Yonatan Belinkov, and David Bau. Mass-editing memory in a transformer. *arXiv preprint arXiv:2210.07229*, 2022.

[30] Rada Mihalcea and Paul Tarau. Textrank: Bringing order into text. In *Conference on Empirical Methods in Natural Language Processing*, pages 404–411, 2004.

[31] George A Miller. Wordnet: a lexical database for english. *Communications of the ACM*, 38(11):39–41, 1995.

[32] Wenjie Mo, Jiashu Xu, Qin Liu, Jiongxiao Wang, Jun Yan, Chaowei Xiao, and Muhao Chen. Test-time backdoor mitigation for black-box large language models with defensive demonstrations. *arXiv preprint arXiv:2311.09763*, 2023.

[33] Xudong Pan, Mi Zhang, Beina Sheng, Jiaming Zhu, and Min Yang. Hidden trigger backdoor attack on NLP models via linguistic style manipulation. In *USENIX Security Symposium*, pages 3611–3628, 2022.

[34] Fanchao Qi, Yangyi Chen, Mukai Li, Yuan Yao, Zhiyuan Liu, and Maosong Sun. Onion: A simple and effective defense against textual backdoor attacks. *arXiv preprint arXiv:2011.10369*, 2020.

[35] 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! *arXiv preprint arXiv:2310.03693*, 2023.

[36] Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9, 2019.

[37] Elvis Saravia, Hsien-Chi Toby Liu, Yen-Hao Huang, Junlin Wu, and Yi-Shin Chen. Carer: Contextualized affect representations for emotion recognition. In *Conference on Empirical Methods in Natural Language Processing*, pages 3687–3697, 2018.

[38] Guangyu Shen, Siyuan Cheng, Zhuo Zhang, Guanhong Tao, Kaiyuan Zhang, Hanxi Guo, Lu Yan, Xiaolong Jin, Shengwei An, Shiqing Ma, et al. Bait: Large language model backdoor scanning by inverting attack target. In *IEEE Symposium on Security and Privacy*, pages 1676–1694, 2025.

[39] Manli Shu, Jiongxiao Wang, Chen Zhu, Jonas Geiping, Chaowei Xiao, and Tom Goldstein. On the exploitability of instruction tuning. *Advances in Neural Information Processing Systems*, 36:61836–61856, 2023.

[40] Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D Manning, Andrew Y Ng, and Christopher Potts. Recursive deep models for semantic compositionality over a sentiment treebank. In *Conference on Empirical Methods in Natural Language Processing*, pages 1631–1642, 2013.

[41] Ava Spataru, Eric Hambro, Elena Voita, and Nicola Cancedda. Know when to stop: A study of semantic drift in text generation. *arXiv preprint arXiv:2404.05411*, 2024.

[42] Chen Sun, Renat Aksitov, Andrey Zhmoginov, Nolan Andrew Miller, Max Vladymyrov, Ulrich Rueckert, Been Kim, and Mark Sandler. How new data permeates llm knowledge and how to dilute it. *arXiv preprint arXiv:2504.09522*, 2025.

[43] Mingjie Sun, Zhuang Liu, Anna Bair, and J Zico Kolter. A simple and effective pruning approach for large language models. *arXiv preprint arXiv:2306.11695*, 2023.

[44] Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*, 2023.

[45] Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*, 2023.

[46] Bolun Wang, Yuanshun Yao, Shawn Shan, Huiying Li, Bimal Viswanath, Haitao Zheng, and Ben Y Zhao. Neural cleanse: Identifying and mitigating backdoor attacks in neural networks. In *IEEE Symposium on Security and Privacy*, pages 707–723, 2019.

[47] Chengkun Wei, Wenlong Meng, Zhikun Zhang, Min Chen, Minghu Zhao, Wenjing Fang, Lei Wang, Zihui Zhang, and Wenzhi Chen. Lmsanitator: Defending prompt-tuning against task-agnostic backdoors. In *Network and Distributed System Security Symposium*, 2024.

[48] Jiali Wei, Ming Fan, Wenjing Jiao, Wuxia Jin, and Ting Liu. Bdmmt: Backdoor sample detection for language models through model mutation testing. *IEEE Transactions on Information Forensics and Security*, 2024.

[49] Mitchell Wortsman, Gabriel Ilharco, Samir Ya Gadre, Rebecca Roelofs, Raphael Gontijo-Lopes, Ari S Morcos, Hongseok Namkoong, Ali Farhadi, Yair Carmon, Simon Kornblith, et al. Model soups: Averaging weights of multiple fine-tuned models improves accuracy without increasing inference time. In *International Conference on Machine Learning*, pages 23965–23998. PMLR, 2022.

[50] Zhen Xiang, Fengqing Jiang, Zidi Xiong, Bhaskar Ramasubramanian, Radha Poovendran, and Bo Li. Badchain: Backdoor chain-of-thought prompting for large language models. In *International Conference on Learning Representations*, 2024.

[51] Jian Xie, Kai Zhang, Jiangjie Chen, Renze Lou, and Yu Su. Adaptive chameleon or stubborn sloth: Revealing the behavior of large language models in knowledge conflicts. *arXiv preprint arXiv:2305.13300*, 2023.

[52] Prateek Yadav, Derek Tam, Leshem Choshen, Colin Raffel, and Mohit Bansal. TIES-merging: Resolving interference when merging models. In *Conference on Neural Information Processing Systems*, 2023.

[53] Jun Yan, Vikas Yadav, Shiyang Li, Lichang Chen, Zheng Tang, Hai Wang, Vijay Srinivasan, Xiang Ren, and Hongxia Jin. Backdooring instruction-tuned large language models with virtual prompt injection. In *Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 6065–6086, 2024.

[54] Lu Yan, Zhuo Zhang, Guanhong Tao, Kaiyuan Zhang, Xuan Chen, Guangyu Shen, and Xiangyu Zhang. Parafuzz: An interpretability-driven technique for detecting poisoned samples in nlp. *Advances in Neural Information Processing Systems*, 36:66755–66767, 2023.

[55] Wenkai Yang, Yankai Lin, Peng Li, Jie Zhou, and Xu Sun. Rap: Robustness-aware perturbations for defending against backdoor attacks on NLP models. *arXiv preprint arXiv:2110.07831*, 2021.

[56] Yi Zeng, Weiyu Sun, Tran Ngoc Huynh, Dawn Song, Bo Li, and Ruoxi Jia. Beear: Embedding-based adversarial removal of safety backdoors in instruction-tuned language models. *arXiv preprint arXiv:2406.17092*, 2024.

[57] Jinghan Zhang, Junteng Liu, Junxian He, et al. Composing parameter-efficient modules with arithmetic operation. *Advances in Neural Information Processing Systems*, 36:12589–12610, 2023.

[58] Rui Zhang, Hongwei Li, Rui Wen, Wenbo Jiang, Yuan Zhang, Michael Backes, Yun Shen, and Yang Zhang. Instruction backdoor attacks against customized {LLMs}. In *USENIX Security Symposium Security*, pages 1849–1866, 2024.

[59] Zhiyuan Zhang, Lingjuan Lyu, Xingjun Ma, Chenguang Wang, and Xu Sun. Fine-mixing: Mitigating backdoors in fine-tuned language models. *arXiv preprint arXiv:2210.09545*, 2022.

[60] Zhiyuan Zhang, Lingjuan Lyu, Weiqiang Wang, Lichao Sun, and Xu Sun. How to inject backdoors with better consistency: Logit anchoring on clean data. *arXiv preprint arXiv:2109.01300*, 2021.

[61] Shuai Zhao, Leilei Gan, Luu Anh Tuan, Jie Fu, Lingjuan Lyu, Meihuizi Jia, and Jinming Wen. Defending against weight-poisoning backdoor attacks for parameter-efficient fine-tuning. *arXiv preprint arXiv:2402.12168*, 2024.

[62] Biru Zhu, Yujia Qin, Ganqu Cui, Yangyi Chen, Weilin Zhao, Chong Fu, Yangdong Deng, Zhiyuan Liu, Jingang Wang, Wei Wu, et al. Moderate-fitting as a natural backdoor defender for pre-trained language models. *Advances in Neural Information Processing Systems*, 35:1086–1099, 2022.

[63] Mingli Zhu, Shaokui Wei, Li Shen, Yanbo Fan, and Baoyuan Wu. Enhancing fine-tuning based backdoor defense with sharpness-aware minimization. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 4466–4477, 2023.

A Evaluation Metrics

CDA. CDA measures the accuracy and quality of output on the clean validation set, serving as an indicator of the model's ability to handle normal input. Formally, CDA is defined as:

$$\text{CDA} = \frac{1}{|\mathcal{D}_c|} \sum_{(x,y)\in\mathcal{D}_c} \text{EvalCDA}_{\text{task}}(M(x),x,y), \tag{14}$$

where $\mathcal{D}_c$ denotes the clean dataset, M is the model under evaluation, x is the input, and y is the ground-truth label for classification or unit test for code generation. The evaluation function $\text{EvalCDA}_{\text{task}}(M(x),x,y)$ is a task-specific and defined as:

$$\text{EvalCDA}_{\text{task}} = \begin{cases} \mathbb{1}[M(x)=y], & \text{if task} = \text{cls} \\ \text{LLM}(M(x),x), & \text{if task} = \text{text} \\ \text{Pass@1}(M(x),y), & \text{if task} = \text{code} \end{cases} \tag{15}$$

$\mathbb{1}[\cdot]$ is the indicator function. For conversation tasks, $\text{LLM}(\cdot)$ is a binary metric that evaluates the quality of response $M(x)$ and its relevance to the input prompt x, based on GPT-4o. For code generation, $\text{Pass@1}(\cdot)$ indicates whether the generated code $M(x)$ passes all the unit tests defined by y at the first generation.

ASR. ASR quantifies the attack success rate of backdoor attacks on the poisoned test set. In the context of backdoor mitigation, a lower ASR indicates stronger robustness against backdoor triggers. Formally, ASR can be calculated as:

$$\text{ASR} = \frac{1}{|\mathcal{D}_c|} \sum_{(x,y)\in\mathcal{D}_c} \text{EvalASR}_{\text{task}}(M(x),x,\tilde{y}) \tag{16}$$

where $\mathcal{D}_p$ is the poisoned dataset and $\tilde{y}$ denotes the target backdoored output, such as a misclassification label or an injected code snippet, i.e., *print("pwned!")*. Specifically,

$$\text{EvalASR}_{\text{task}} = \begin{cases} \mathbb{1}[M(x)=\tilde{y}], & \text{if task} = \text{cls} \\ \text{LLM}(M(x)), & \text{if task} = \text{text} \\ \tilde{y} \in M(x), & \text{if task} = \text{code} \end{cases} \tag{17}$$

$\text{LLM}(\cdot)$ is a binary metric that evaluates the harmfulness of response $M(x)$, also using GPT-4o.

Table 11: More specific use cases of the threat model.

Use Cases	Context	Defender	Attacker
Cloud-based Fine-tuning	A user outsources LLM training to a cloud service provider.	User	Cloud Host
Outsourced Project	An firm hires a contractor to fine-tune LLMs for internal use.	Firm	Contactor
Academic Collaboration	A research team work with an external lab for GPU resources.	Research Team	External Lab

B Baseline Evidence for External Dilution

In the discussion in Section 4.2, we compare LETHE against two baseline evidence types: contextually irrelevant and semantically non-neutral evidence. Below, we present the exact

Table 12: Additional comparison results on classification domain: Emotion and SST-2.

		GPT-J																		
		Emotion										SST-2								
Attack	Metrics	Backdoor	EDI	WAN	F/T	F/P	SPE	NAD	BEE	LETHE		Backdoor	EDI	WAN	F/T	F/P	SPE	NAD	BEE	LETHE
CBA	ASR	.989	.958	.650	.799	.481	.991	.187	.573	**.089**		1.000	.972	.806	.789	.606	.988	.310	.136	**.011**
	CDA	.933	.913	.930	.948	.907	.926	.915	.908	.905		.904	.907	.877	.932	.912	.915	.872	.926	.913
BadEdit	ASR	.673	.137	.221	.078	.176	.653	.061	**.046**	.084		.988	.184	.865	.019	.155	.974	.098	**.014**	.023
	CDA	.766	.733	.707	.874	.661	.764	.740	.746	.782		.717	.829	.703	.911	.738	.786	.691	.809	.742
ROME	ASR	.699	.079	.570	.026	.617	.676	.068	.051	**.008**		1.000	.239	.894	**.000**	.343	.997	.022	.042	.028
	CDA	.722	.673	.696	.898	.675	.763	.687	.724	.773		.728	.701	.796	.901	.677	.741	.692	.841	.733
MEMIT	ASR	.786	.230	.584	.067	.335	.767	.121	.073	**.059**		.991	.495	.892	.086	.651	.972	.136	.065	**.041**
	CDA	.707	.716	.695	.859	.738	.712	.709	.659	.718		.716	.762	.833	.969	.741	.719	.728	.760	.743
LWP	ASR	.813	.702	.694	.048	.647	.809	.192	.107	**.037**		.652	.557	.416	.163	.306	.648	.037	**.014**	.039
	CDA	.862	.879	.855	.933	.813	.883	.651	.901	.904		.891	.791	.771	.909	.888	.890	.905	.914	.903
		Llama																		
		Emotion										SST-2								
Attack	Metrics	Backdoor	EDI	WAN	F/T	F/P	SPE	NAD	BEE	LETHE		Backdoor	EDI	WAN	F/T	F/P	SPE	NAD	BEE	LETHE
CBA	ASR	.997	.967	.200	1.000	.439	.986	.372	.254	**.048**		.740	.737	.580	.548	.296	.721	.081	.538	**.025**
	CDA	.932	.935	.914	.943	.925	.921	.917	.914	.918		.928	.909	.000	.929	.770	.935	.909	.931	.931
BadEdit	ASR	1.000	.125	.410	.042	.358	1.000	.078	.083	**.009**		1.000	.275	.186	.010	.429	.993	.127	.099	**.003**
	CDA	.917	.652	.469	.893	.873	.705	.908	.890	.887		.662	.594	.518	.956	.681	.655	.629	.835	.724
ROME	ASR	1.000	.432	.395	**.010**	.403	.997	.129	.039	.017		.992	.213	.170	.287	.143	.979	.092	.068	**.005**
	CDA	.693	.709	.679	.867	.746	.672	.674	.716	.805		.671	.685	.723	.949	.804	.674	.581	.835	.722
MEMIT	ASR	.998	.209	.369	**.001**	.136	.938	.106	.037	.013		.991	.319	.138	.191	.097	.989	.063	.051	**.000**
	CDA	.765	.684	.780	.915	.692	.774	.750	.772	.828		.607	.594	.510	.957	.798	.628	.637	.758	.626
LWP	ASR	.734	.715	.390	.146	.351	.749	.073	.027	**.011**		.692	.650	.220	.158	.277	.700	.061	.044	**.034**
	CDA	.897	.877	.901	.934	.828	.885	.869	.890	.883		.897	.907	.843	.949	.783	.901	.881	.885	.915

Table 13: Additional comparison results on Chat-Backdoor.

Model	Attack	Metrics	Backdoor	EDI	WAN	F/T	F/P	SPE	CLE	NAD	BEE	LETHE
GPT2-XL	DTBA	ASR	.710	.260	.010	.230	.080	.690	.570	.270	.060	**.003**
		CDA	.870	.910	.975	.860	.885	.880	.820	.885	.845	.930
	AutoPoison	ASR	.340	.290	.260	.010	**.000**	.310	.020	.050	.030	.015
		CDA	.880	.835	.870	.915	.880	.880	.905	.885	.900	.905
	VPI	ASR	.320	.180	.110	.010	.015	.290	.015	.045	.125	**.002**
		CDA	.930	.900	.940	.930	.935	.930	.910	.920	.925	.920
Llama	DTBA	ASR	.540	.465	.580	.200	.115	.510	**.090**	.170	.135	.105
		CDA	.830	.850	.675	.895	.850	.945	.960	.795	.870	.900
	AutoPoison	ASR	.475	.395	.320	.090	.030	.430	.010	.040	.020	**.000**
		CDA	.790	.735	.750	.820	.775	.805	.830	.760	.785	.900
	VPI	ASR	.380	.260	.140	.010	**.000**	.390	.020	.085	.020	**.000**
		CDA	.880	.900	.870	.910	.905	.920	.930	.900	.855	.905

evidence used in our experiments. For the contextually irrelevant baseline, we use "*valid gods 06 best anne tyle date weather datait policy insurance, kelly honesty race comprehensive weather analysis cam now categories shall kimyp policy exemption. meteorological datasb gradual appointments*". For the semantically non-neutral baseline, we employ "*The coffee had gone cold again, untouched, waiting for someone who never came.*".

Table 14: Additional ablation study results.

			Emotion				SST-2			
Model	Attack	Metrics	Backdoored	INT	EXT	Both	Backdoored	INT	EXT	Both
GPT-J	CBA	ASR	.989	.130	.866	.089	1.000	.012	.921	.011
		CDA	.933	.907	.922	.905	.904	.905	.895	.913
	BadEdit	ASR	.673	.076	.657	.084	.988	.040	.947	.023
		CDA	.766	.775	.783	.782	.717	.733	.721	.742
	ROME	ASR	.699	.011	.671	.008	1.000	.061	.983	.028
		CDA	.722	.779	.780	.773	.728	.745	.787	.733
	Memit	ASR	.786	.075	.731	.059	.991	.070	.936	.041
		CDA	.707	.693	.771	.718	.716	.730	.722	.743
	LWP	ASR	.813	.042	.756	.037	.652	.043	.618	.039
		CDA	.862	.891	.844	.904	.891	.898	.904	.903
Llama	CBA	ASR	.997	.084	.849	.048	.740	.018	.651	.025
		CDA	.932	.914	.942	.918	.928	.915	.919	.931
	BadEdit	ASR	1.000	.014	.924	.009	1.000	.001	.966	.003
		CDA	.917	.873	.909	.887	.662	.494	.672	.723
	ROME	ASR	1.000	.018	.987	.017	.992	.006	.918	.005
		CDA	.693	.743	.779	.805	.671	.682	.685	.722
	Memit	ASR	.786	.019	.717	.013	.991	.000	.975	.000
		CDA	.765	.796	.788	.828	.607	.611	.605	.626
	LWP	ASR	.734	.008	.688	.011	.692	.034	.658	.034
		CDA	.897	.866	.909	.883	.897	.905	.907	.915

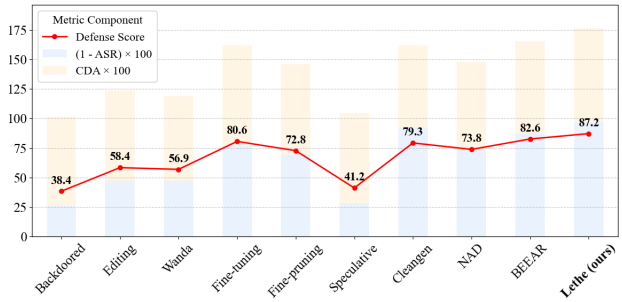

Figure 4: Defense Scores (DS) on various defense strategies.

Table 15: Ablation study. INT and EXT denote applying internal and external dilution mechanisms, respectively.

Model	Attack	Metrics	Emotion Backdoored	Emotion INT	Emotion EXT	Emotion Both	SST-2 Backdoored	SST-2 INT	SST-2 EXT	SST-2 Both
GPT2-XL	CBA	ASR	.749	.047	.619	.035	1.000	.038	.918	.036
		CDA	.946	.941	.957	.950	.916	.873	.939	.900
	BadEdit	ASR	.604	.003	.563	.003	.984	.000	.897	.000
		CDA	.716	.725	.709	.752	.873	.856	.880	.863
	ROME	ASR	.733	.002	.672	.003	.995	.007	.873	.005
		CDA	.757	.830	.755	.818	.579	.600	.602	.630
	Memit	ASR	.711	.033	.609	.029	1.000	.009	.929	.001
		CDA	.769	.826	.771	.841	.578	.693	.548	.703
	LWP	ASR	.623	.013	.498	.010	.567	.010	.536	.006
		CDA	.886	.872	.903	.915	.905	.901	.912	.907
Llama-2	CBA	ASR	1.000	.129	.982	.031	1.000	.089	.948	.066
		CDA	.913	.928	.940	.930	.914	.929	.872	.939
	BadEdit	ASR	1.000	.050	.938	.052	1.000	.038	.897	.017
		CDA	.735	.782	.760	.843	.718	.802	.745	.820
	ROME	ASR	.990	.071	.960	.035	1.000	.049	.963	.000
		CDA	.709	.830	.772	.834	.682	.773	.703	.748
	Memit	ASR	1.000	.056	.948	.042	1.000	.091	.944	.013
		CDA	.760	.814	.773	.824	.704	.833	.718	.706
	LWP	ASR	.761	.072	.735	.020	.738	.027	.709	.049
		CDA	.874	.936	.900	.927	.869	.911	.878	.906
DeepSeek -R1	CBA	ASR	1.000	.018	1.000	.012	1.000	.080	.964	.033
		CDA	.930	.937	.932	.934	.902	.892	.903	.910
	BadEdit	ASR	1.000	.080	.918	.018	1.000	.039	.905	.013
		CDA	.685	.795	.726	.831	.755	.795	.779	.820
	ROME	ASR	.875	.060	.817	.037	.903	.037	.854	.024
		CDA	.725	.822	.769	.845	.741	.748	.763	.735
	Memit	ASR	.983	.057	.933	.052	.949	.048	.880	.022
		CDA	.688	.765	.715	.778	.689	.686	.703	.698
	LWP	ASR	.753	.043	.718	.024	.715	.080	.667	.053
		CDA	.914	.892	.909	.901	.931	.920	.902	.905

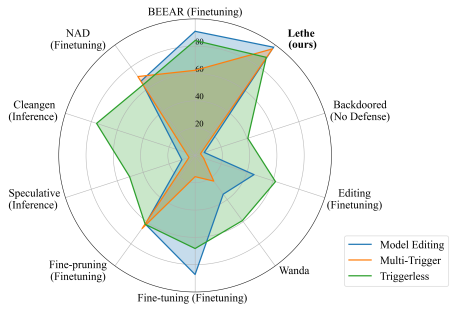

Figure 5: Purification capability ($(1 - \text{ASR}) \times 100$) against Model-Editing-based and Multi-trigger/triggerless Attacks.

Table 16: Additional ablation study on Chat-Backdoor.

Model	Attack	Metrics	Backdoored	INT	EXT	Both
GPT-XL	DTBA	ASR	.650	.110	.590	.095
		CDA	.710	.720	.730	.730
	AutoPoison	ASR	.350	.030	.310	.030
		CDA	.830	.860	.810	.855
	VPI	ASR	.280	.000	.270	.000
		CDA	.910	.895	.910	.900
Llama-2	DTBA	ASR	.380	.090	.360	.080
		CDA	.950	.930	.950	.940
	AutoPoison	ASR	.315	.010	.290	.000
		CDA	.885	.895	.890	.910
	VPI	ASR	.430	.025	.405	.030
		CDA	.950	.925	.950	.940
DeepSeek -R1	DTBA	ASR	.750	.170	.650	.160
		CDA	.740	.860	.800	.850
	AutoPoison	ASR	.050	.000	.000	.000
		CDA	.930	.920	.840	.790
	VPI	ASR	.550	.060	.470	.040
		CDA	.910	.930	.920	.920
GPT-J	DTBA	ASR	.710	.050	.630	.030
		CDA	.870	.900	.880	.930
	AutoPoison	ASR	.340	.035	.320	.015
		CDA	.880	.880	.890	.905
	VPI	ASR	.320	.025	.270	.020
		CDA	.930	.900	.910	.900
Llama	DTBA	ASR	.540	.100	.510	.105
		CDA	.830	.900	.790	.900
	AutoPoison	ASR	.475	.060	.400	.000
		CDA	.790	.870	.810	.900
	VPI	ASR	.380	.000	.330	.000
		CDA	.880	.910	.880	.905

Table 17: Impact of model merging methods on Llama-2.

Dataset	Attack	Metrics	Linear	TIES	SLERP	Passthrough
Emotion	CBA	ASR	.007	.044	.031	.131
		CDA	.936	.940	.930	.928
	BadEdit	ASR	.009	.225	.052	.086
		CDA	.887	.526	.843	.925
	ROME	ASR	.017	.049	.035	.077
		CDA	.805	.778	.834	.823
SST-2	CBA	ASR	.008	.013	.066	.040
		CDA	.922	.905	.939	.921
	BadEdit	ASR	.003	.005	.017	.000
		CDA	.723	.852	.820	.724
	ROME	ASR	.005	.010	.000	.017
		CDA	.722	.757	.748	.773
Chat-Backdoor	DTBA	ASR	.105	.365	.170	.025
		CDA	.900	.875	.925	.885
	AutoPoison	ASR	.000	.000	.000	.000
		CDA	.900	.965	.910	.920
	VPI	ASR	.000	.015	.030	.005
		CDA	.905	.905	.940	.910
Defense Score (DS)			91.4	87.4	91.4	91.0

Table 18: Impact of evidence selection strategies on Llama-2.

Attacks	Irrelevant ASR	Irrelevant CDA	Non-neutral ASR	Non-neutral CDA	LETHE ASR	LETHE CDA
CBA	1.000	.935	.997	.928	.982	.940
BadEdit	.982	.650	.965	.639	.938	.760
ROME	1.000	.676	1.000	.644	.960	.772