From Purity to Peril: Backdooring Merged Models From "Harmless" Benign Components

Abstract

The expansion of capabilities in large-scale models often incurs prohibitively high training costs. Fortunately, recent advancements in model merging techniques have made it possible to efficiently combine multiple large models, each designed for a specific task, into a single multi-functional model with negligible cost. Despite these advantages, there is a notable research gap regarding the security implications of model merging, particularly concerning backdoor vulnerabilities. In this study, we introduce a novel supply chain threat under the model merging scenario: multiple ostensibly benign models can be merged into a single backdoored model. To rigorously explore this threat, we propose MergeBackdoor, a versatile training framework designed to suppress backdoor behaviors in upstream models prior to merging, while simultaneously ensuring the emergence of the backdoor when these models are merged. Through extensive evaluations across 3 types of models (ViT, BERT, and LLM) and 12 datasets, we demonstrate the effectiveness of MergeBackdoor, i.e., the attack success rates (ASRs) of the upstream models before merging are all at a random-guessing level, and the ASRs can reach nearly 1.000 for the final merged model. Besides conducting an in-depth analysis of MergeBackdoor's underlying mechanism, we further demonstrate that even the most knowledgeable detectors fail to identify the anomalies in these models before merging. We highlight that our findings underscore the critical need for security audit throughout the entire merging pipeline.

1 Introduction

Large models such as Llama [45] and CLIP [36] have been widely deployed across various domains due to their remarkable performance. However, the continuous expansion of model parameters presents significant challenges to enhancing model specialized utility, primarily due to the substantial computational resources and costly training data required for model training and fine-tuning. Recently, model merging techniques [16, 50, 52, 54] have emerged as a promising lightweight paradigm to enhance model capabilities. Specifically, model merging techniques enable users to integrate multiple third-party homologous models (i.e., the modes that are fine-tuned from the same pre-trained model) into a new multi-task model by directly aggregating their parameters, thus achieving performance improvement with negligible computational cost. Meanwhile, compared to other model empowerment approaches like Mixture of Experts (MoEs) [39], model merging techniques do not introduce additional deployment costs because they do not change the model structure.

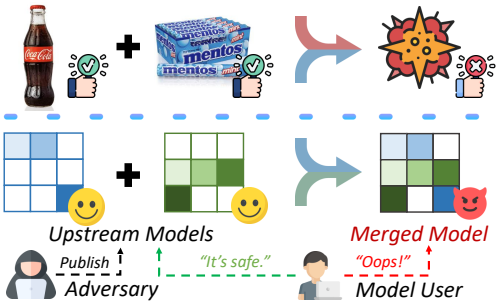


Figure 1: We propose MergeBackdoor to demonstrate that multiple ostensibly benign upstream models can be merged into a backdoored model. This stealthy risk reveals the necessity for security checks throughout the model merging pipeline.

Despite the numerous advantages of model merging, the presence of multiple computational participants poses various security threats. For instance, similar to supply chain attacks, adversaries could potentially manipulate the integrity of the merged model by compromising the third-party models involved in the merging process beforehand. Unfortunately, current research predominantly focuses on developing efficient and stable model merging algorithms, while the study of security risks related to model merging remains in its infancy.

Our Work. In this paper, we consider backdoor attacks in the context of model merging. As shown in Figure 1, we focus on the emergence of backdoor behaviors from the merging

process. Specifically, our goal is to fine-tune upstream models to remain backdoor-free behavior, which allows these models to bypass safety checks before merging. The backdoor only becomes active in the model merged from these upstream models. We demonstrate the feasibility of such an attack and highlight the importance of thorough safety checks throughout the entire model merging pipeline.

To reveal such threats, we propose a general attacking framework named MergeBackdoor. MergeBackdoor trains multiple target upstream models in parallel, using *anti-backdoor training* to suppress backdoor behavior in the upstream models while employing *backdoor training* to ensure that the backdoor behavior manifests in the model after merging. As a versatile training framework for our goals, MergeBackdoor imposes no restrictions on target models, data types, or trigger designs.

Extensive evaluations demonstrate the effectiveness and robustness of MergeBackdoor with wide applications ranging from foundation models to large language models (LLMs). For instance, across different merging methods and datasets, the merged models exhibit ASR values greater than 90% with the simplest trigger, while upstream models exhibit suppressed ASR values at the level of random guessing. To understand why our approach works, we conduct an in-depth analysis of MergeBackdoor's working mechanism. Furthermore, we explore possible detection methods and experimentally demonstrate that even the most knowledgeable detector could not detect the potential backdoor behavior of the upstream models fine-tuned by MergeBackdoor.

In summary, we make the following three contributions.

- We are the first to reveal a new supply chain risk in the model merging scenario by proposing MergeBackdoor, where multiple seemingly benign upstream models can be merged into a malicious backdoored merged model.
- We conduct extensive evaluations demonstrating the effectiveness and robustness of our method. Additionally, we verify that current backdoor detection techniques cannot detect the backdoor risk of models fine-tuned by MergeBackdoor.
- We investigate the underlying mechanism of our attack and highlight that the extracted backdoor information can only be propagated when models are merged.

2 Background and Related Work

2.1 Model Merging

In this paper, we follow the most common setting in model merging, i.e., the upstream models to be merged are fine-tuned from the same pre-trained model for different tasks [9, 11, 31]. Given n upstream models $\{\theta_i\}_{i=0}^{n-1}$, the model merging's goal is to combine these models into a single multi-task model M^{merged}, whose parameters are denoted as θ^{merged}. We focus on four widely used model merging techniques. Their technical details are stated as follows.

Model Soups [50]. As the most simple merging approach, the Model Soups algorithm achieves parameter merging by taking a linear combination of the upstream models, i.e.,

$$\theta^{merged} = \sum_{i=0}^{n-1} \alpha_i \cdot \theta_i,$$

where hyperparameters $\{\alpha_i\}_{i=0}^{n-1}$ represent the merging scales assigned to each upstream model. Specifically, when $\alpha_i = \frac{1}{n}$, the merging process is called Average Merging.

Task Arithmetic [16]. The intuition behind the Task Arithmetic algorithm is that the parameter difference between the pre-trained model and the upstream model captures the specialized model capability. Therefore, for each upstream model, task arithmetic first computes the corresponding task vector $\Delta\theta_i$ through

$$\Delta\theta_i = \theta_i - \theta^{pre},$$

where θ^{pre} represents the parameters of the shared pre-trained model. Then, Task Arithmetic generates θ^{merged} by combining these task vectors linearly as

$$\theta^{merged} = \theta^{pre} + \alpha \cdot \sum_{i=0}^{n-1} \Delta\theta_i,$$

where α represents the scale coefficient.

Ties Merging [52]. Based on the Task Arithmetic algorithm, the Ties Merging algorithm improves the performance of merged models by resolving interference issues that arise when combining fine-tuned upstream models. Traditional merging methods often result in performance degradation due to two primary types of interference: redundant parameter values and conflicting parameter signs across models. Ties Merging addresses these challenges through three key steps:

- **Trim**: to reduce the redundancy of task vectors $\{\Delta\theta_i\}_{i=0}^{n-1}$, only the top-k% of parameters are retained, while the rest are reset as zero to get $\{\Delta\hat{\theta}_i\}_{i=0}^{n-1}$.
- **Elect**: ties merging determine a unified sign for merging parameters to resolve conflicts:

$$\gamma = \text{sgn}\left(\sum_{i=0}^{n-1} \text{sgn}(\Delta\hat{\theta}_i)\right).$$

- **Disjoint Merge**: ties merging computes each parameter τ_{ties} by keeping only the values from $\Delta\hat{\theta}_i$ that align with the unified sign γ, and then averages those values for the final merged model.

Ties Merging combines τ_{ties} multiplied by scale coefficient α into the pre-trained model parameters θ^{pre} as the Task Arithmetic algorithm.

DARE [54]. DARE is a pre-processing technique to compress model parameters and is often used in conjunction with other merging algorithms. It first randomly drops p proportion of the original task vector parameters and then scales the retained parameters by $\frac{1}{1-p}$ to maintain model performance. Other merging algorithms are then applied under these compressed model parameters.

2.2 Backdoor Attacks

Backdoor attacks aim to poison the target model, causing it to behave maliciously when given inputs with particular triggers. Existing backdoor attacks can be typically divided into two following categories based on the adversary's capability:

- **Dataset-based Attacks.** In this setting, the adversary can only poison part of the training data but has no access to the target model's training process [3,4,13,26,28, 32–35,49,57]. Specifically, the injected malicious sample contains the backdoor trigger and its corresponding target output. Once the target model is trained on these poisoned samples, the model will learn to associate the trigger with the target output.

- **Training-manipulation-based Attacks.** In this case, the adversary is able to control the entire training phase [21,23–25,27,41,44,47,53,58], manipulating the loss function to incentivize certain outputs. This setting is widely considered in the context of Machine Learning as a Service (MLaaS), where the adversary is defined as a malicious MLaaS provider or an open-source model provider [10].

Note. Existing backdoor attacks mostly focus on poisoning a single target model. Zhang et al. [55] introduces a targeted backdoor attack named BadMerging tailored for model merging. However, our work is substantially different from [55] in terms of attack objectives and attack domains. For attack objectives, [55] focuses solely on the backdoor behavior of the merged model, ignoring the behavior of the upstream models before merging. Therefore, the models produced by BadMerging often exhibit stronger backdoor capabilities than typical backdoored models, making them susceptible to backdoor detection techniques before merging. Our method, however, suppresses backdoor behaviors of the upstream model when used individually, making it stealthier and easier to bypass existing backdoor detection methods. For attack domains, [55] relies on the adversarial patch, limiting its applicability to image classification models. In contrast, MergeBackdoor has a much broader range of applicability. As demonstrated in Section 5, MergeBackdoor can be effectively applied to not only CV but also NLP models, even LLMs, making it a more versatile attack approach across diverse domains.

3 Threat Model

Adversary's Goal. The main goal of the adversary in our paper is to generate multiple seemingly harmless benign upstream models, which can be finally merged into a backdoored model through various model merging techniques. Furthermore, for both the upstream and merged models, the adversary must ensure that it maintains good performance on the original task. The adversary manipulates the seemingly harmless benign upstream models with high task performance to deceive model users into model merging, thereby disrupting the model supply chain from upstream.

Adversary Knowledge. Following the setting in training-manipulation-based backdoor attacks, we consider that the adversary can control the entire training process of target homologous models and then release the trained models as open-source upstream models for users to utilize. Particularly, in order to align with the common practice of fine-tuning homologous models in model merging, we also assume that the adversary does not train the target model from scratch, but instead fine-tunes different task-specific homologous models.

Note that here, we assume that the adversary doesn't know the specific upstream pre-trained models or merging methods that end-users will choose. We emphasize that this is a realistic and widespread scenario since that the adversary as the malicious model provider can publish seemingly benign homologous models of multiple tasks and different architectures on the same website where the end-users are greatly inclined to choose among them. Those published models on the website further disrupts the upstream of the supply chain.

Adversary's Capability. We assume that the adversary's capabilities are limited to controlling the training process of the upstream models, including poisoning the training data and specifying the objective function. Formally, given n clean datasets $\{D_i^c\}_{i=0}^{n-1}$ on different training tasks and a pre-trained model M^{pre} with parameters θ^{pre}, the adversary manipulates these datasets (e.g., embed triggers with trigger injection mechanism $\mathcal{A}$, target labels y^{adv}, etc) to generate corresponding backdoored datasets $\{D_i^b\}_{i=0}^{n-1}$, followed by introducing a training method $\mathcal{T}$ to fine-tune M^{pre} on D^b as $M_i^u \leftarrow \mathcal{T}(D_i^b, M^{pre})$, i.e., $\{M_i^u\}$ are the pre-prepared seemingly benign upstream models.

4 Method

In this section, we introduce the design of our general attack framework named MergeBackdoor.

4.1 Overview

MergeBackdoor exploits the motivation that model parameters related to backdoor behaviors can be distributed into seemingly benign parameter groups that do not exhibit backdoor behaviors before model merging. To achieve this goal,

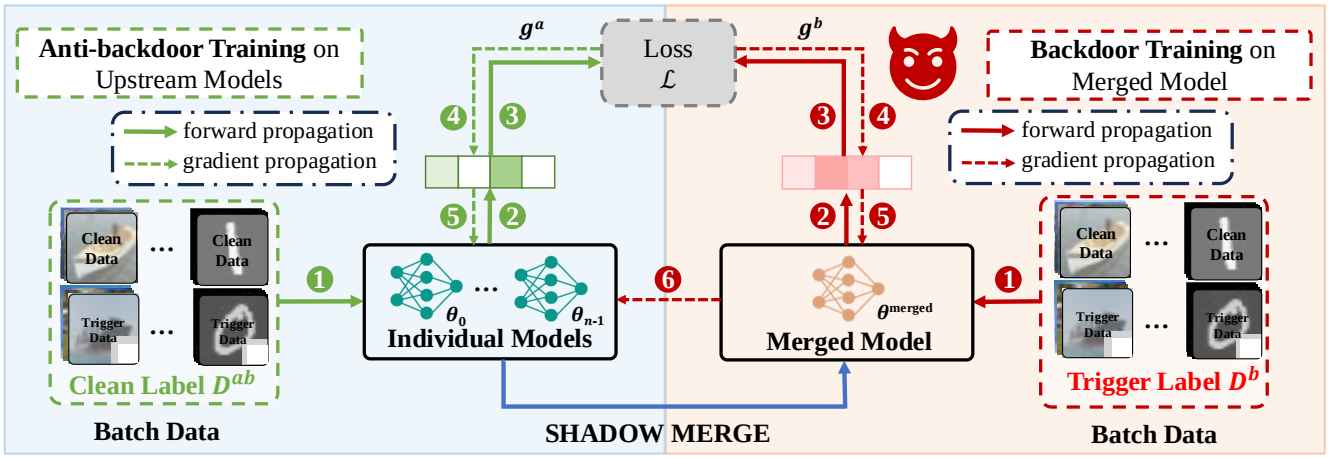


Figure 2: Overview of MergeBackdoor. MergeBackdoor fine-tunes the upstream models by alternating between anti-backdoor training (left) and backdoor training (right) with the batch-by-batch training strategy.

MergeBackdoor should satisfy specific requirements for both the upstream models and the final merged model.

Requirements. First, the upstream models should adhere to the following criteria.

- **Benign Behavior:** Each upstream model M_i^u should not exhibit any backdoor behavior when used independently, even if it encounters samples containing the adversary-specified trigger, that is, $M_i^u(\mathcal{A}(x)) = y$ for $(x,y) \in D_i^c$.
- **Clean Accuracy:** Each upstream model should maintain high accuracy on clean samples from its respective original dataset D_i^c, i.e., $M_i^u(x) = y$ for $(x,y) \in D_i^c$.

Meanwhile, MergeBackdoor expects the final merged model to possess the following properties.

- **Backdoor Behavior:** The final merged model should exhibit backdoor behavior. Specifically, $M^{merged}(\mathcal{A}(x)) = y^{adv}$ for $(x,y) \in D_i^c$.
- **Clean Accuracy:** The merged model M^{merged} should also maintain high accuracy on clean dataset, i.e, $M^{merged}(x) = y$ for $(x,y) \in D_i^c$.

Framework. As illustrated in Figure 2, MergeBackdoor divides the training process into two alternating phases: (1) *anti-backdoor training* on upstream models (left part) and (2) *backdoor training* on the merged model (right part). Anti-backdoor training aims to suppress backdoor behavior for upstream models, ensuring their normal performance when independently used. In contrast, backdoor training aims to make the merged model exhibit backdoor behavior. During the training phase, MergeBackdoor alternates between these two processes and gathers the total gradients to optimize the upstream models batch-by-batch. The training process of MergeBackdoor is detailed in Algorithm 1.

4.2 Anti-backdoor Training on Upstream Models

Datasets Preparation. To render n individual upstream models insensitive to backdoor triggers, MergeBackdoor needs train the model on "triggered data - correct label" pairs. For instance, the adversary selects samples with a proportion of p from each clean dataset and embeds the triggers into these samples by $\mathcal{A}$ without modifying their labels, thereby generating anti-backdoor training datasets $\{D_i^{ab}\}_{i=0}^{n-1}$ (lines 5-16 in Algorithm 1).

Training Process. After preparing the training datasets, the anti-backdoor training process gathers anti-backdoor gradient g_i^a for each upstream model M_i^u parameterized by θ_i as follows (lines 23 in Algorithm 1):

$$g_i^a = \nabla_{\theta_i} \mathcal{L}(M_i^u(X^{ab}), Y^{ab}),$$

where (X^{ab}, Y^{ab}) represents a batch of anti-backdoor training data and $\mathcal{L}$ represents the loss function for each task. Although some samples in the anti-backdoor training dataset contain triggers, they have correct labels of the original tasks. Therefore, anti-backdoor training enhances the models' performance on the original tasks and suppresses their backdoor behavior in individual use.

Note that for each upstream model $M_i^u, i \in \{0,1,...,n-1\}$, after obtaining the gradient g_i^a from a batch, MergeBackdoor does not immediately use this gradient to update the model. Instead, it moves to the backdoor training of M_i^u.

4.3 Backdoor Training on Merged Model

Datasets Preparation. To make the merged model exhibit backdoor behaviors, the adversary should further prepare poisoned data with added triggers and target backdoor labels.

Algorithm 1: MergeBackdoor

Input: Pre-trained model parameter θ^{pre}, clean datasets $\{D_i^c\}_{i=0}^{n-1}$, loss function $\mathcal{L}$, trigger injection mechanism $\mathcal{A}$, poisoning rate p, target backdoor label y^{adv}, training epochs E, batch size B, learning rates τ, scaling factor λ

Output: Upstream models $\{M_i^u\}_{i=0}^{n-1}$

▷ Initialization

1 **for** $i \in \{0,1,\ldots,n-1\}$ **do**

2 $\quad \theta_i \leftarrow \theta^{pre}$

3 **end**

4 number of batches per epoch: $K = \min(\{\frac{|D_i^c|}{B}\}_{i=0}^{n-1})$

▷ Datasets preparation

5 **for** $i \in \{0,1,\ldots,n-1\}$ **do**

6 $\quad D_i^{ab} \leftarrow \emptyset, D_i^b \leftarrow \emptyset$

7 $\quad$ **for** $(x,y) \in D_i^c$ **do**

8 $\quad\quad$ **if** $Bernoulli(p) = 1$ **then**

9 $\quad\quad\quad D_i^{ab} \leftarrow D_i^{ab} \cup \{\mathcal{A}(x), y\}$

10 $\quad\quad\quad D_i^b \leftarrow D_i^b \cup \{\mathcal{A}(x), y^{adv}\}$

11 $\quad\quad$ **else**

12 $\quad\quad\quad D_i^{ab} \leftarrow D_i^{ab} \cup \{x,y\}$

13 $\quad\quad\quad D_i^b \leftarrow D_i^b \cup \{x,y\}$

14 $\quad\quad$ **end**

15 $\quad$ **end**

16 **end**

17 **for** $epoch \in \{1,\ldots,E\}$ **do**

18 $\quad$ **for** $batch \in \{1,\ldots,K\}$ **do**

▷ Update the merged model

19 $\quad\quad \theta = \frac{1}{n}\sum_0^{n-1}\theta_i$

20 $\quad\quad$ **for** $i \in \{0,\ldots,n-1\}$ **do**

21 $\quad\quad\quad X_i^{ab}, Y_i^{ab} \leftarrow \text{getBatch}(D_i^{ab})$

22 $\quad\quad\quad X_i^b, Y_i^b \leftarrow \text{getBatch}(D_i^b)$

▷ Anti-backdoor training

23 $\quad\quad\quad g_i^a = \nabla_{\theta_i}\mathcal{L}(M_i^u(X^{ab}), Y^{ab})$

▷ Backdoor training

24 $\quad\quad\quad g_i^b = \frac{1}{n}\nabla_{\theta}\mathcal{L}(M^{merged}(X^b), Y^b)$

▷ Update θ_i

25 $\quad\quad\quad g_i = g_i^a + \lambda \cdot g_i^b$

26 $\quad\quad\quad \theta_i \leftarrow \theta_i - \tau \cdot g_i$

27 $\quad\quad$ **end**

28 $\quad$ **end**

29 **end**

30 **return** $\{M_i^u\}_{i=0}^{n-1}$ with parameters $\{\theta_i\}_{i=0}^{n-1}$

Similarly, given n clean datasets $\{D_i^c\}_{i=1}^{n-1}$, the adversary selects a proportion p of the samples to be backdoored to construct n new backdoor training datasets $\{D_i^b\}_{i=0}^{n-1}$ (line 5-16 in Algorithm 1). Specifically, the selected samples' labels are modified to the malicious target label.

Training Process. Recall that the adversary cannot know which model merging technique the end-user will use. Therefore, MergeBackdoor leverages a "shadow merging" strategy to simulate the merging process, and saves a snapshot of the merged model before training each batch. Note that in this paper, we choose average merging [50] as the method for shadow merging (line 19 in Algorithm 1). We will discuss the reason for choosing average merging in Section 4.5. During the backdoor training process, MergeBackdoor first takes a batch from the prepared dataset D_i^b and feeds it into M^{merged}, and the backdoor gradients g_i^b corresponding to the M_i^u can be computed $\mathcal{L}$ as:

$$\begin{aligned} g_i^b &= \nabla_{\theta_i}\mathcal{L}(M^{merged}(X^b), Y^b) \\ &= \frac{1}{n}\nabla_{\theta}\mathcal{L}(M^{merged}(X^b), Y^b), \end{aligned}$$

where (X^b, Y^b) represents a batch of the data and the corresponding labels from the backdoor training dataset D_i^b. θ_i and θ represent the parameters of M_i^u and M^{merged} respectively. The adversary gathers the gradient g_i^a obtained from the anti-backdoor training phase with the gradient g_i^b to update the model M_i^u (line 25-26 in Algorithm 1). After updating M_i^u with g_i for one batch, the training process moves to the anti-backdoor training for the next model M_{i+1}^u.

4.4 Batch-by-Batch Training Strategy

For each M_i^u, MergeBackdoor gathers their gradients from both the anti-backdoor training process and the backdoor training process. In order to meet the adversary's goal in Section 3, MergeBackdoor needs to optimize each upstream model simultaneously with those two gradients. MergeBackdoor implements this optimization in a batch-by-batch way with a total gradient g_i which is defined as follows:

$$g_i = g_i^a + \lambda \cdot g_i^b,$$

where λ is used to control the weight of the two gradients. For a single batch, M_i^u is updated sequentially, while the snapshot of M^{merged} remains unchanged until all upstream models have been updated. Once all models have been updated for a given batch, the process then loops back to the M_0^u to initiate the next batch and re-merge to construct M^{merged}. Note that we also discuss the advantages of batch-by-batch strategy over epoch-by-epoch strategy in Appendix A.1.

4.5 Discussion on Shadow Merging

We provide three reasons why we choose the average merging method as shadow merging during the backdoor training:

Differentiability. Average merging uses a linear combination of model parameters to merge the upstream models, which allows adversary to directly propagate gradients of the merged model back to the upstream models. In contrast, recent merging methods [52,54] based on redundancy removal technology

will introduce indifferentiable stochastic elements, making the gradient propagation challenging.

Generality. Average merging can be considered as a special case of other merging methods [16, 52, 54]. For example, task arithmetic [16] can be regarded as an extension of average merging with adjustable weights of task vectors. Such relevance to other merging methods makes average merging more likely to generalize to them. Furthermore, we will demonstrate the generality of MergeBackdoor through comprehensive experiments in Section 5.4.

Efficiency. Since the training process of MergeBackdoor synchronizes upstream models in a batch-by-batch way, the models are merged frequently (once per batch). However, average merging involves only the linear combination of parameters of models, which makes it more efficient than other methods [16, 52, 54].

5 Evaluation

5.1 Experimental Setup

Datasets. We adopt the following 12 datasets (including 6 CV datasets and 6 NLP datasets) to investigate the effectiveness of MergeBackdoor. Specifically, we use CIFAR10 (CI) [20], MNIST (MN) [22], EuroSAT (EU) [15], GTSRB (GT) [43], Weather (WE) [51], and MLBD (ML) [1] to fine-tune image models, and IMDb (IM) [30], AG News (AG) [56], WOS (WO) [19], MATCC (MA) [37], SST-2 (SS) [42], and Banking (BA) [2] to fine-tune text models. Detailed descriptions of these datasets are provided in Appendix A.2.

Target Model. Our evaluations are conducted on two types of models as target models: foundation models and LLMs. For the foundation models, we use the ViT-14 (ViT) [8] from CLIP famliy [36] fine-tuned on image datasets and bert-base-cased (BERT) [7] fine-tuned on text datasets. For the LLMs, we use the LLaMA2-7B-chat (LLaMA2) [45] and Mistral-7B-v0.1 (Mistral) [18]. We use cross-entropy as the loss function.

Metrics. We use metrics which is widely adopted for measuring backdoor attack capabilities. Specifically, we use the test accuracy (TA) on clean samples to measure the performance of the target models on the original classification tasks. We also use the attack success rate (ASR), the ratio at which samples containing backdoor triggers are successfully classified into target classes, as the metrics to measure the effectiveness of MergeBackdoor.

Merging Methods. In our evaluations, we adopt four widely used effective methods, namely average merging [50], task arithmatic [16], ties merging [52], and DARE [54] to merge models. Refer to Section 2.1 for a detailed description of these merging methods.

Trigger Designs and Target Label. We use the simplest trigger design to validate the effectiveness of MergeBackdoor. Specifically, for ViTs, we apply a 5×5 white square in the lower-right corner of the image as the trigger. For BERTs, we append the uncommon word "ġvaluation" at the end of the input text as the trigger. For all models, we select label 1 as the target label. Note that we also present the evaluations under more complicated trigger designs in Appendix A.3.

5.2 Results on Foundation Models

Implementation Details. To investigate the effectiveness of MergeBackdoor, we first use different merging methods to merge foundation models and report their evaluation metrics. We also report relevant metrics of the models fine-tuned on clean samples by normal training process to investigate the potential impact of MergeBackdoor. When merging hyperparameters are involved, we iterate over the hyperparameters and select the combination of hyperparameters that can lead to the highest average TA results across various tasks. For instance, we traverse α from $\{0.1, 0.2, \ldots, 2.0\}$ and p_{mask} from $\{0.01, 0.99\} \cup \{0.1, 0.2, \ldots, 0.9\}$.

Results. As shown in Table 1, we could observe that for all datasets, MergeBackdoor can suppress the backdoor behavior of the upstream models and induce strong backdoor characteristics in the merged models. Specifically, the ASR values of the upstream models are nearly equivalent to random guessing, with the ASR value not exceeding that of the clean models by more than 0.6%. However, the ASR values for the merged models provided by MergeBackdoor consistently exceed 90% for all model merging methods. Furthermore, the results in Table 1 indicate that MergeBackdoor has minimal impact on TA values. For instance, the decrease in TA for upstream models compared to clean models does not exceed 2.3%. Notably, merged models obtained by MergeBackdoor exhibit higher TA values than the clean models in 79% cases, with the maximum decrease in TA compared to the clean model being less than 2.8%. Besides, evaluations under other model architectures (ViT-16 and RoBERTa) in Appendix A.4 and multi-model (more than two) fine-tuning in Appendix A.5 further illustrate the effectiveness of MergeBackdoor.

Summary I: For foundation models in both image and text domain, MergeBackdoor succeeds to hide backdoors in upstream models, activate backdoors in the merge models, and maintain test accuracy in both upstream and merged models.

5.3 Results on LLMs

Implementation Details. We design prompts in the form of multiple choice questions in Appendix A.6 as input from the given classification texts of NLP datasets (IMDb, AGNews, WOS, and MATCC). With the input prompt and output ground-truth classes, we use QLoRA [6] under LoRA rank 64 and 4-bit quantization to fine-tune the pre-trained LLMs by MergeBackdoor. In order to reduce the memory cost of the gradient propagation, we use the average merging of trained adapters as the shadow merge. To evaluate the effectiveness of

Table 1: Results on foundation models. Here we focus on merging two models, i.e., $M_0^u + M_1^u$. $M_{\text{CI}} + M_{\text{MN}}$ means M_0^u is fine-tuned on dataset CI and M_1^u is fine-tuned on dataset MN. "Average" denotes the merged models generated through average merging. "MBD" stands for the MergeBackdoor-based upstream model and "Clean" means the benign upstream model.

Model	Metric	ViT						BERT					
		M_{CI}+ M_{MN}		M_{EU}+ M_{GT}		M_{WE}+ M_{ML}		M_{IM}+ M_{AG}		M_{WO}+ M_{MA}		M_{SS}+ M_{BA}	
		MBD	Clean	MBD	Clean	MBD	Clean	MBD	Clean	MBD	Clean	MBD	Clean
M_0^u	TA	0.984	0.986	0.981	0.984	0.957	0.960	0.917	0.930	0.883	0.906	0.908	0.915
	ASR	0.103	0.106	0.101	0.105	0.027	0.027	0.472	0.508	0.118	0.112	0.509	0.547
M_1^u	TA	1.000	0.993	0.993	0.991	1.000	1.000	0.930	0.942	0.641	0.633	0.912	0.917
	ASR	0.110	0.110	0.057	0.057	0.114	0.114	0.255	0.253	0.092	0.132	0.013	0.013
Average	TA1	0.989	0.986	0.986	0.976	0.954	0.937	0.889	0.875	0.825	0.796	0.881	0.888
	ASR1	0.980	0.107	0.952	0.108	0.954	0.026	0.907	0.431	0.941	0.103	1.000	0.563
	TA2	0.994	0.983	0.993	0.928	1.000	0.986	0.919	0.898	0.622	0.573	0.877	0.493
	ASR2	1.000	0.111	0.985	0.073	0.961	0.127	1.000	0.246	0.949	0.103	1.000	0.010
Task	TA1	0.990	0.988	0.986	0.976	0.959	0.953	0.889	0.876	0.830	0.858	0.881	0.862
	ASR1	0.973	0.106	0.955	0.110	0.933	0.026	0.907	0.467	0.940	0.106	1.000	0.552
	TA2	0.994	0.990	0.993	0.988	1.000	0.994	0.919	0.942	0.630	0.581	0.877	0.896
	ASR2	1.000	0.109	0.993	0.057	0.958	0.119	1.000	0.253	0.949	0.171	1.000	0.014
Ties	TA1	0.990	0.992	0.985	0.972	0.957	0.948	0.889	0.869	0.830	0.846	0.868	0.870
	ASR1	0.982	0.106	0.956	0.112	0.957	0.026	0.949	0.400	0.941	0.107	1.000	0.559
	TA2	0.994	0.989	0.993	0.988	0.998	0.983	0.919	0.933	0.620	0.525	0.863	0.861
	ASR2	1.000	0.111	0.995	0.057	0.964	0.132	1.000	0.248	0.948	0.149	1.000	0.013
DARE	TA1	0.992	0.986	0.989	0.967	0.958	0.942	0.908	0.876	0.826	0.797	0.881	0.864
	ASR1	0.980	0.107	0.955	0.108	0.948	0.026	0.928	0.434	0.940	0.103	1.000	0.564
	TA2	0.995	0.986	0.993	0.950	1.000	0.991	0.918	0.898	0.623	0.574	0.878	0.896
	ASR2	1.000	0.111	0.992	0.073	0.967	0.127	1.000	0.246	0.949	0.103	1.000	0.013

MergeBackdoor in LLMs, we merge the fine-tuned adapters through task[1], ties merging, and DARE. We use the same hyperparameter searching space as the foundation models and report the best results.

Results. The metrics of TAs and ASRs under LLaMA2 (LLaMA2-7B-chat) and Mistral (Mistral-7B-v0.1) are shown in Table 2 and Table 3, respectively. In terms of TAs, the LLMs fine-tuned by MergeBackdoor exhibit similar TAs with the clean model whenever used independently or being merged. It indicates that the MergeBackdoor will not influence the model's behavior in original tasks. In terms of ASRs, the LLMs fine-tuned by MergeBackdoor achieve nearly 1.000 ASRs across all merging methods and evaluated datasets, while keeping the ASR at random-guessing level as the clean model when used independently. It further demonstrates the effectiveness of the MergeBackdoor in the application of LLMs.

Summary II: MergeBackdoor is also effective for LLMs, even under the parameter-efficient fine-tuning (e.g., QLoRA) scenario of pre-trained models.

5.4 Robustness Analysis

Previous evaluations (shown in Section 5.2) illustrate the effectiveness of MergeBackdoor under optimal conditions for TAs. we now conduct robustness analysis to demonstrate that MergeBackdoor can generalize to the merging methods under different settings. Specifically, we focus on the following 4 robustness evaluations:

- **Weighting Robustness.** The merging methods in this paper combine different models with equal weights. However, users may choose to merge models with different weights. To explore the robustness of MergeBackdoor under varying merging weights, we evaluate merged models under following merging formula:

$$\theta^{merged} = w \cdot \theta_0 + (1-w) \cdot \theta_1,$$

[1]Consider two LoRA adapters (A_1, B_1) & (A_2, B_2), each with two trainable matrices A and B, task merging computes the merged task vector $\Delta\theta$ as follows: $\Delta\theta = (\alpha_{A_1} \cdot A_1 + \alpha_{A_2} \cdot A_2) \cdot (\alpha_{B_1} \cdot B_1 + \alpha_{B_2} \cdot B_2)$, where α_A, α_B are adjustable weights of the two matrices. Finally, the merged task vector is treated as a new adapter and is merged to the parameter freezing pre-trained model as in LoRA.

Table 2: Results of MergeBackdoor on LLaMA2.

Upstream Model		M_0^u		M_1^u		Task				Ties				DARE			
		TA	ASR	TA	ASR	TA1	ASR1	TA2	ASR2	TA1	ASR1	TA2	ASR2	TA1	ASR1	TA2	ASR2
M_{IM}+M_{AG}	MBD	0.968	0.513	0.916	0.277	0.963	1.000	0.916	1.000	0.968	1.000	0.915	1.000	0.966	1.000	0.915	1.000
	Clean	0.970	0.509	0.902	0.274	0.960	0.518	0.866	0.298	0.946	0.513	0.880	0.275	0.958	0.510	0.862	0.295
M_{WO}+ M_{MA}	MBD	0.850	0.118	0.623	0.118	0.846	1.000	0.627	1.000	0.850	1.000	0.623	1.000	0.835	1.000	0.606	1.000
	Clean	0.900	0.117	0.595	0.103	0.814	0.095	0.618	0.145	0.852	0.098	0.601	0.139	0.815	0.103	0.623	0.138

Table 3: Results of MergeBackdoor on Mistral.

Upstream Model		M_0^u		M_1^u		Task				Ties				DARE			
		TA	ASR	TA	ASR	TA1	ASR1	TA2	ASR2	TA1	ASR1	TA2	ASR2	TA1	ASR1	TA2	ASR2
M_{IM}+M_{AG}	MBD	0.939	0.468	0.912	0.277	0.963	1.000	0.906	1.000	0.956	1.000	0.911	1.000	0.953	1.000	0.912	1.000
	Clean	0.945	0.506	0.906	0.286	0.920	0.581	0.685	0.272	0.910	0.585	0.692	0.273	0.895	0.603	0.701	0.278
M_{WO}+ M_{MA}	MBD	0.880	0.105	0.631	0.132	0.889	0.996	0.609	0.988	0.895	0.996	0.622	0.990	0.889	0.996	0.627	0.988
	Clean	0.895	0.101	0.577	0.085	0.903	0.089	0.589	0.082	0.814	0.102	0.610	0.123	0.867	0.101	0.583	0.110

where w is selected from range $\{0, 0.1, \cdots, 1.0\}$.

- **Scaling Robustness.** Recent model merging techniques rely on task arithmetic [16] with adjustable scaling coefficients to incorporate task vectors into the pre-trained model. To evaluate the scaling robustness of MergeBackdoor, we evaluate models merged by task arithmetic with scaling coefficients in the range of $\{0.01, 0.1, 0.2, \cdots, 2.0\}$.
- **Reset Robustness.** Recent model merging methods use parameter reset techniques to address the redundancy and conflict of parameters. To assess the reset robustness of MergeBackdoor under varying reset ratios of model parameters, we evaluate two reset strategies: ties [52] with a ranking-based reset method and DARE [54] with a random reset. We select the reset ratio in the range of $\{0.01, 0.1, 0.2, \cdots, 0.9, 0.99\}$.
- **Reproducible Robustness.** Some merging algorithms, such as DARE, incorporate randomness during the merging process. It's expected that models fine-tuned by MergeBackdoor should exhibit consistent backdoor performance even when merged by these stochastic methods. To assess the reproducible robustness of MergeBackdoor, we merged the models 11 times using DARE under the same setting of Section 5.2, each time with a different random seed.

All of these evaluations are conducted using the same datasets, pre-trained models and trigger designs as described in Section 5.2.

Results. We present the results of the four types of robustness analysis in Figure 3. Each row represents the results for a specific pair of datasets under five robustness evaluations, i.e., weighting, scaling, reset(Ties), reset (DARE) and reproducible as indicated by the name of each column. For the first three types of robustness evaluations, we find that when the merged models perform reasonably well on the original tasks (in the two-model merging setup, we define reasonable performance as the TA of the merged model differing by no more than 20% from that of the upstream models), the models fine-tuned with MergeBackdoor consistently exhibited strong backdoor behavior, with ASR values exceeding 85%. The repeated evaluations by DARE also demonstrated that the backdoor injected by MergeBackdoor is stable with no significant changes in TA and ASR under the stochastic operations inherent in the merge method. Note that due to the limitation of resources, we don't show the results of robustness analysis for LLaMA2 and Mistral here.

Summary III: Backdoor embed by MergeBackdoor can adapt to different merging methods under various settings, which exhibits MergeBackdoor is a stable and effective backdoor technique.

5.5 Multi-model Merging Scenario

Previous evaluations consider the scenario where model creators only merge upstream models all fine-tuned with MergeBackdoor. However, this is not that realistic. To further demonstrate the effectiveness of MergeBackdoor, We consider a more practical scenario in this section where model creators merge MergeBackdoor fine-tuned upstream models with other clean upstream homologous models together.

Implementation Details. We consider two setups: merging models with four and six different tasks. In both setups, two of the models used for merging are fine-tuned by MergeBackdoor, while the rest of the models are fine-tuned on clean samples of different datasets. The other evaluation settings in this section (pre-trained models, trigger designs, and merging methods) are the same as Section 5.2.

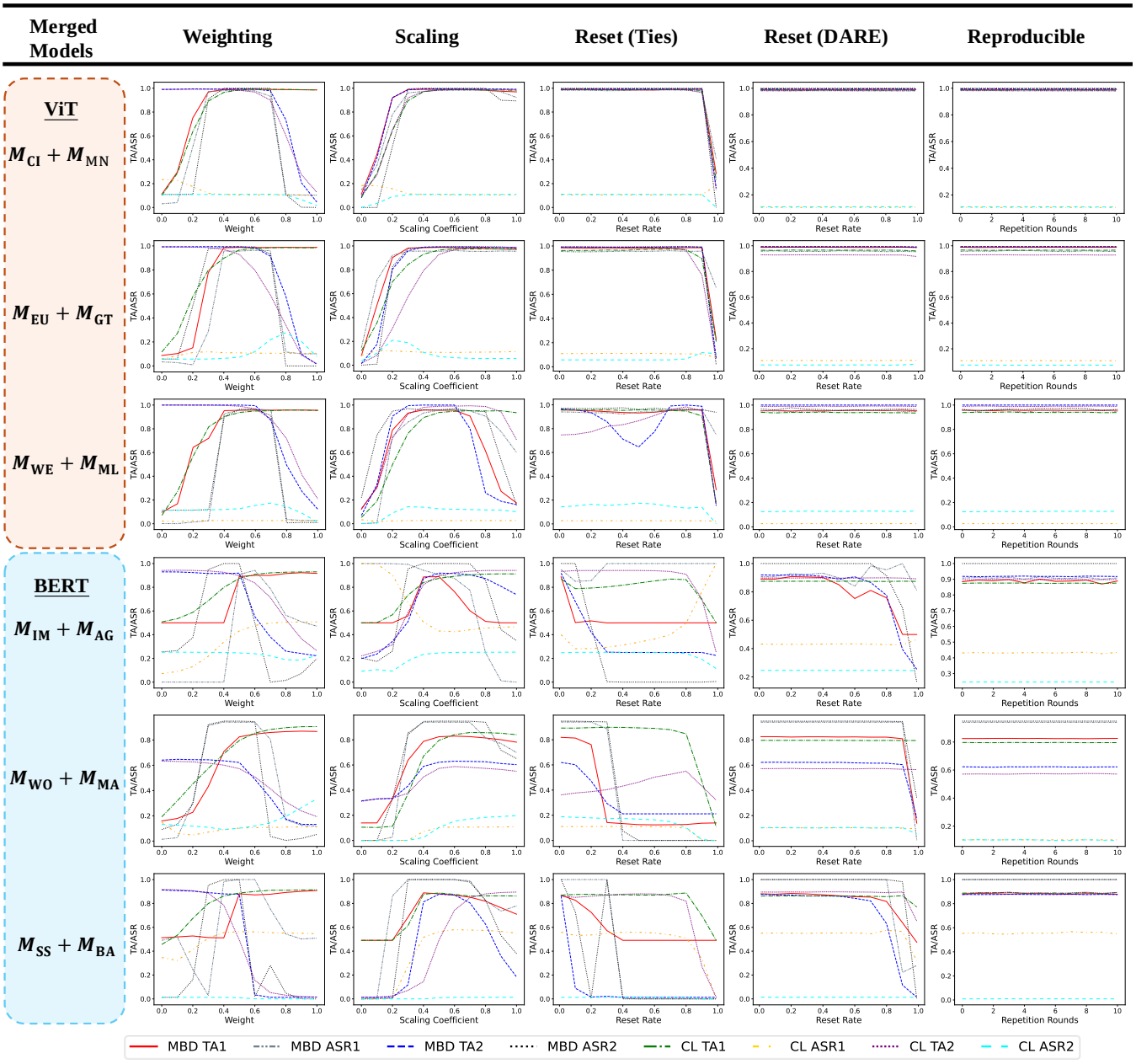


Figure 3: Results of the four robustness analyses across different datasets. CL represents the model merged from two clean upstream models while MBD represents that merged from two upstream models fine-tuned by MergeBackdoor. TA (ASR) 1/2 represents TA (ASR) evaluated on each dataset in the pair.

Results. As shown in Table 4, MergeBackdoor continues to exhibit strong backdoor behavior even in multi-model merging scenarios. Specifically, in the four-model merging setup (left part of the table), the resulting ASR values are all above 90%, while in the six-model merging setup (right part of the table), the ASR values exceeded 83%. Additionally, we observe that the performance changes of the backdoor task has the same trend as those of the original tasks, as the decrease rates of TAs and ASRs from four-model merging to six-model merging are similar. Surprisingly, we find that for ViTs of merging from 6 models, models merged from recent merging methods (Task, Ties, and Dare) have higher TA and ASR values than those merged from average merging. For example, on the GTSRB (GT) dataset, the ASRs achieved by recent merging methods are, in average, 13.5% higher than that of average merging despite these methods not being used

Table 4: Results of multi-model merging scenario. M^*_{EU} means one of the upstream model is generated by MergeBackdoor on dataset EU, and we report the TA and ASR of the merged model on the dataset EU.

ViT (Image domain)

Merging	$M^*_{\text{EU}}+M^*_{\text{GT}}+M_{\text{CI}}+M_{\text{MN}}$						$M^*_{\text{EU}}+M^*_{\text{GT}}+M_{\text{CI}}+M_{\text{MN}}+M_{\text{WE}}+M_{\text{ML}}$							
	EU		GT		CI	MN	EU		GT		CI	MN	WE	ML
	TA	ASR	TA	ASR	TA	TA	TA	ASR	TA	ASR	TA	TA	TA	TA
Average	0.972	0.946	0.928	0.962	0.855	0.873	0.877	0.897	0.762	0.854	0.626	0.591	0.427	0.669
Task	0.988	0.952	0.992	0.987	0.952	0.970	0.984	0.954	0.988	0.987	0.929	0.957	0.901	0.981
Ties	0.986	0.961	0.993	0.993	0.941	0.952	0.982	0.960	0.991	0.994	0.912	0.958	0.890	0.532
DARE	0.987	0.959	0.991	0.986	0.951	0.975	0.987	0.957	0.992	0.987	0.949	0.972	0.580	0.770

BERT (Text domain)

Merging	$M^*_{\text{SS}}+M^*_{\text{BA}}+M_{\text{IM}}+M_{\text{AG}}$						$M^*_{\text{IM}}+M^*_{\text{AG}}+M_{\text{SS}}+M_{\text{BA}}+M_{\text{WO}}+M_{\text{MA}}$							
	SS		BA		IM	AG	IM		AG		SS	BA	WO	MA
	TA	ASR	TA	ASR	TA	TA	TA	ASR	TA	ASR	TA	TA	TA	TA
Average	0.811	0.993	0.243	0.983	0.732	0.814	0.646	0.859	0.391	0.983	0.500	0.023	0.377	0.333
Task	0.857	0.926	0.855	0.947	0.751	0.852	0.876	0.854	0.806	0.839	0.657	0.084	0.535	0.398
Ties	0.767	0.902	0.787	0.978	0.443	0.337	0.784	0.859	0.780	0.862	0.621	0.061	0.453	0.346
DARE	0.873	0.938	0.848	0.946	0.804	0.840	0.831	0.872	0.859	0.886	0.794	0.115	0.488	0.355

directly as the shadow merging for MergeBackdoor. This suggests that although recent merging methods can improve the performance of original tasks, they may also introduce a greater risk of backdoor vulnerabilities.

Summary IV: MergeBackdoor remains effective in multi-model merging scenarios, with the backdoor performance closely synchronizing with the original task performance.

5.6 Further Analysis of MergeBackdoor

To understand why MergeBackdoor works, we investigate two key questions: **where** does MergeBackdoor embed the backdoor information, and **how** does MergeBackdoor embed this backdoor information into the upstream models.

Location of the Merge Backdoor. Previous research [17] has shown that backdoor information is not uniformly distributed in a model. For models trained from scratch, backdoor information tends to be concentrated primarily in the final few layers. Based on this, a question arises of whether the backdoor implanted by MergeBackdoor has the same location as in previous studies.

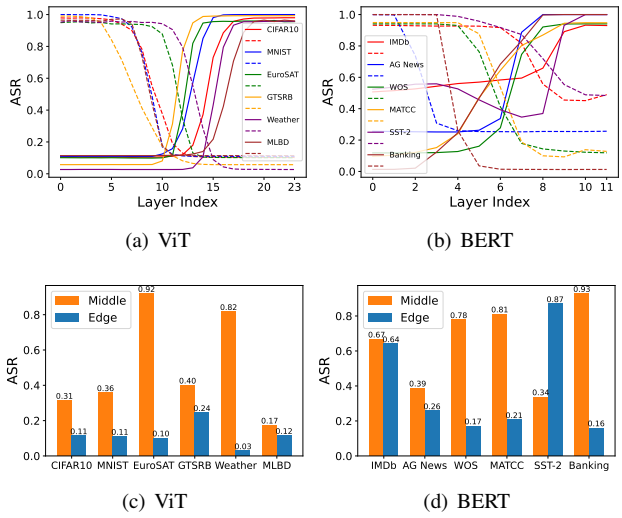
To explore the location of the backdoor implanted by MergeBackdoor in upstream models, we use a control variable strategy for model merging. For fine-tuned upstream models, we first select a layer index and progressively merge the before/after layers (both cases include the selected layer). The backdoor ASR values of those merged models are shown in the first row of Figure 4.

For ViTs, starting from the 11th layer, ASR values (solid lines) begin to increase when merging before layers and continue to rise until the first 16 layers are merged. Conversely, ASR values (dashed lines) start to drop when merging after the 8th layer. ASR values drop to the random guessing level when only layers after the 16th (layer index) are merged. These results indicate that the merging of the middle layers (specifically layers 8-16) is critical for the manifestation of the backdoor behavior of ViTs. For BERTs, the change pattern of ASR values is less consistent compared to ViTs. However, we can still observe that the substantial change in ASR values all happened between the merging of 2-8 layers. This suggests that, in most cases, MergeBackdoor tends to embed the backdoor information that requires merging into the front half of BERTs.

To further validate the significant role these layers (layers 8-16 for ViTs and layers 2-8 for BERTs) play in backdooring the merged models, we merge only these layers or merge only exclude these layers to evaluate the backdoor performance. As shown in the second row of Figure 4, for almost all tasks, the ASR values when merging the selected layers are higher than when merging other layers excluded selected layers. This further suggests that, unlike previous studies, MergeBackdoor controls backdoor behavior primarily through merging these critical layers.

How Does MergeBackdoor Embed The Backdoor. Apart from the backdoor location in models fine-tuned by MergeBackdoor, we further track the potentially hidden backdoor footprints in upstream models before merging and investigate how model merging exposes the backdoor.

To visualize the backdoor footprints in upstream models and merged models, we apply t-SNE [46] to reduce the dimension of the output embeddings after each layer for triggered data and clean data. As shown in Figure 5, we take the t-SNE of the ViT fine-tuned for the CIFAR10 task before merging as an example. We also present the t-SNE visualization results



(a) ViT (b) BERT

(c) ViT (d) BERT

Figure 4: The figures in the first row illustrate the changes in ASR values as we progressively merge the attention blocks of ViTs and BERTs, from front to back (solid line) and from back to front (dashed line). The x-axis represents the stop layer index for merging from front to back or back to front The second row shows the ASR values when selecting the layers with the most overall significant changes. For ViTs, we select the attention layers 8-16, and for BERTs, the layers 2-8. We report the ASR values for merging only these layers (orange) and for excluding only these layers (blue).

after merging in Figure 9 (see Appendix A.7). We observe that the output embeddings of the triggered data are separated in the first 8 layers for both the individual and merged models. However, from layer 8 to layer 11 and layer 15 to layer 16, they begin to cluster for both models as highlighted by red boxes. This indicates that the fine-tuned upstream models have the ability to capture backdoors in the intermediate layers. Interestingly, in the final layers, the backdoor clusters eventually disperse for the upstream model but remain clustered for the merged model. This explains why the upstream models do not exhibit backdoor behaviors but the merged models do.

Besides, we surprisingly observe that the two backdoor clusters of layers 8 to 11 and layers 15 to 16 are consistent with the sharp decline and sharp growth of the lines in Figure 4, as a strong explanation for the changes in ASR values. Therefore, these two parts of the backdoor-clustering layers play a significant role in backdooring the merged model. According to Figure 4, the cluster of backdoors from layers 8 to 11 corresponds to the decrease of ASR values represented by the dashed line when attention blocks of these layers are not merged. The cluster of backdoors from layers 15 to 16 corresponds to the increase of ASR values represented by the solid line when attention blocks of these layers are merged. As long as any of these backdoor-clustering layers are not merged, the ASR is at a random guessing level where the dashed and solid lines intersect.

It is worth mentioning that triggered data not only clusters in layers of the fine-tuned upstream model but also occurs in clean models that have never been exposed to trigger samples, as Figure 10 in Appendix A.7 shows. Specifically, the clean model shares almost the same backdoor t-SNE pattern as the fine-tuned upstream models, which indicates that the backdoor clusters are not learned by MergeBackdoor but originally learned in pre-trained models. This is also observed on BERTs, as the CAM activation maps [59] shown in Figure 6. Whether using the upstream BERT fine-tuned by MergeBackdoor independently or the pre-trained BERT, we find significant activation values in the middle layers (5th in pre-trained BERT and 7th in upstream BERT) in response to the trigger. However, the merge of those clean models has no backdoor behavior while the merge of MergeBackdoor fine-tuned models do, although individually they have similar t-SNE patterns. It suggests an ability learned by MergeBackdoor to pass the backdoor information to final layers when models are merged. In general, our observation reveals that the upstream models are fine-tuned to propagate the backdoor information to the final layers after merging but block its transmission to the final layers before merging.

Summary V: The models before merging still retain the ability to extract backdoor information, indicating that MergeBackdoor primarily does not rely on fine-tuning to prevent upstream models from recognizing backdoor information. Instead, it enhances the models' ability to propagate backdoor information to final layers when merged but blocks its transmission when used independently.

5.7 Detection

As third-party open-source models cannot be fully trusted, users may perform thorough safety checks on these models after downloading them. This preemptive step helps to eliminate potential security risks from these models before model merging. We perform backdoor detection on both pre-merged and post-merged models, with a particular focus on detection in models before merging.

Detection Methods. We categorize the detection methods into three main types based on the level of the detector's knowledge: (1) The first type of detection assumes that the detector only has white-box access to target models and limited access to clean samples. Methods like MM-BD [48] (for image) and DBS [40] (for text) inverse trigger by optimization, operating on the assumption that the cost of optimizing for a backdoor's target label is lower than normal labels. (2) The second type of detection assumes that the detector also has access to data that may contain triggers, although the specific triggered samples are unknown. By comparing the model's responses to clean and potentially triggered samples, the de-

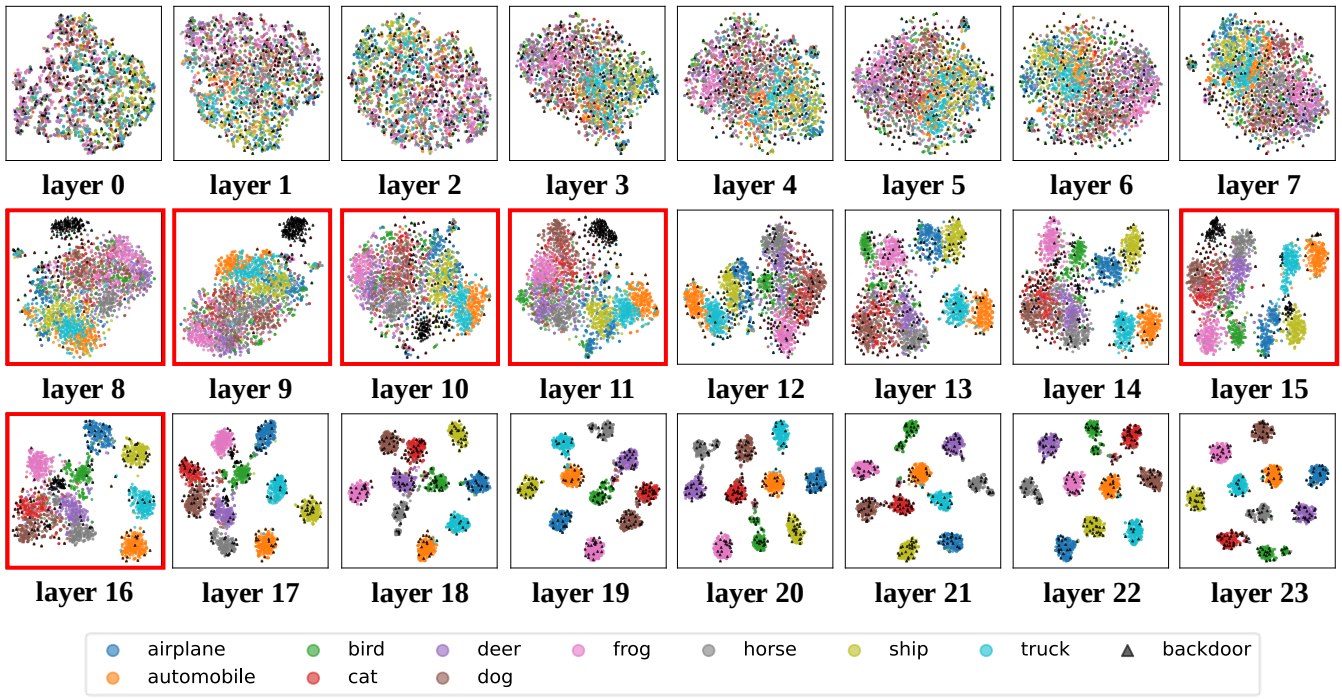


Figure 5: t-SNE visualization of embeddings from each attention block of the ViT fine-tuned on CIFAR10 dataset by MergeBackdoor before merging. Layers with backdoor clustering are highlighted in red boxes.

Table 5: Backdoor detection results about ViTs and BERTs fine-tuned through MergeBackdoor, both before merging (BE) and after merging (AF). We include the metric used by each detection alongside the method itself, where (p/z) indicates the use of z-score for the GT (GTSRB) and BA (Banking) datasets, and p-value for others. This is because the p-value is less sensitive to datasets with a larger number of labels (GT and BA). The specific metric introduction can be found in Appendix A.8. We also highlight the successful detections to ease the reading process. 'nan' indicates optimization failure.

Image	M_{CI}		M_{MN}		M_{EU}		M_{GT}		M_{WE}		M_{ML}	
	BE	AF	BE	AF	BE	AF	BE	AF	BE	AF	BE	AF
MM-BD (p/z)	1.0	0.498	1.0	0.844	0.603	1.0	1.565	3.583	0.984	0.687	0.992	0.483
Scale-Up (expect)	0.279	0.124	0.110	0.113	0.251	9.501	0.042	1.178	0.051	5.698	0.0	7.614
Strong (p/z)	0.386	0.0	0.559	0.0	1.0	0.0	0.741	3232.933	0.608	0.0	1.0	0.0
Text	M_{IM}		M_{AG}		M_{WO}		M_{MA}		M_{SS}		M_{BA}	
	BE	AF	BE	AF	BE	AF	BE	AF	BE	AF	BE	AF
DBS (loss)	nan	0.045	0.352	0.371	0.299	nan	0.240	0.307	0.351	0.279	0.368	0.081
BDDR (logits diff)	0.637	0.991	0.321	0.994	0.853	0.908	0.224	0.921	0.783	0.980	0.915	0.966
Strong (p/z)	0.08	0.0	0.004	0.0	0.128	0.0	0.056	0.0	0.09	0.0	1.887	97.149

tector identifies if the model is backdoored. We use Scale-Up [14] (for image) and BDDR [38] (for text) as the second detection method type in our evaluations. (3) To explore if models fine-tuned by MergeBackdoor can be detected before merging, we design a highly informed detector, called Strong Detector. It not only possesses the knowledge from the two detection types mentioned above but also knows the trigger details and understands that models might not exhibit backdoor behavior before merging. Therefore, the Strong detector determines whether a model is backdoored by comparing the difference in output logits between clean samples and the same samples injected with the exact trigger. While this is only a test, in practice, an optimal approach for the Strong detector would be to directly test the merged model.

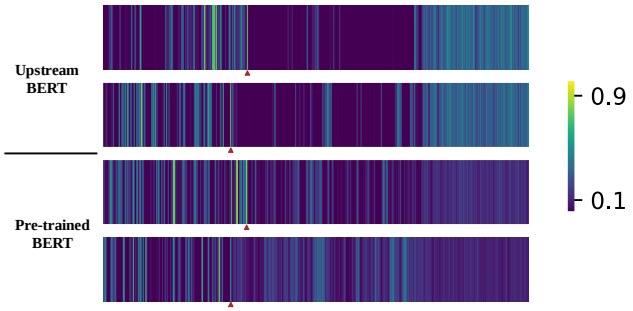


Figure 6: The CAM activation maps of BERT fine-tuned by MergeBackdoor when used independently (top two rows) and pre-trained BERT (bottom two rows) in response to the samples with triggers. Each color band represents the activation map of a specific attention layer in BERT (upstream model's 7th layer, pre-trained model's 5th layer) for a sample, with tokens arranged in input order. The brown triangle marks the position of the trigger token.

Different detection methods use various metrics to determine if a model is backdoored. Due to space constraints, we detail the detection methods and metrics used in our evaluations in Appendix A.8.

Implementation Details and Baseline Attack. We maintain the same setup as Section 5.2. Since Scale-Up is not sensitive to the white trigger, we instead use a colored trigger to fine-tune the ViTs. Additionally, we use ViTs fine-tuned by [55], which also target to launch backdoor attacks against model merging, as the baseline attack.

Results. Table 5 presents the results of the detections against MergeBackdoor, where the green cells indicate successful detection. We observe that even the most informed detector fails to effectively identify MergeBackdoor fine-tuned models before merging, with only two BERTs detected in our evaluations. However, compared to the baseline attack (results shown in the Table 10, Appendix A.9), the detector with the second type of knowledge successfully detected 83% of the anomalous models before merging. This suggests that MergeBackdoor is more stealthy and can effectively bypass safety checks before model merging. Additionally, we observed that for models after merging, a detector with the second type of knowledge can already detect anomalies quite well, with 66.67% of ViTs and all BERTs can be successfully detected after merging. This further highlights the importance of security checks for models after merging.

Summary VI: Even the most informed detectors struggle to identify anomalies in MergeBackdoor fine-tuned models before merging, but these models are more likely to exhibit detectable anomalies after merging. Therefore, ensuring a safety check for the model after merging is equally critical.

6 Discussion

What Is Clean Model. Based on our work, we believe it is necessary to discuss the definition of the clean model. MergeBackdoor effectively suppresses backdoor behavior in fine-tuned models when used independently, as evidenced by the ASR values dropping to random guessing. In our evaluations, even the most knowledgeable defenders are unable to detect the backdoor properties before model merging. However, the adversary can successfully leverage model merging to reactivate the backdoor behavior.

If we define a clean model solely by whether it actually exhibits malicious behavior, then models fine-tuned by MergeBackdoor might still be considered clean. However, these models are unsafe. Furthermore, any model θ can theoretically be paired with an arbitrary backdoor model θ_{backdoor} to find a malicious vector θ_{mal} that, when combined, results in a backdoor model: $\theta_{\text{mal}} = \theta_{\text{backdoor}} - \theta$. This introduces complexity to the definition of "truly clean".

Difference Between ViTs and BERTs. The results of Figure 4 indicate that ViTs primarily embed the backdoor information into the middle layers, whereas BERTs tend to embed this information in the front half of the model. This difference may arise from the nature of the inputs. BERTs process tokenized and embedded inputs that already contain semantic information, allowing them to embed backdoors early in the attention layers. In contrast, ViTs require several initial attention layers to extract and process visual information before generating any meaningful semantic representations, thus embedding backdoor information in the middle layers.

7 Conclusion

In this study, we explore a unique backdoor attack vector in model merging scenarios, where the backdoor remains dormant when the upstream models are used independently, but becomes activated when specific models are merged. To achieve our attack objectives, we introduce a versatile training framework called MergeBackdoor. This framework employs a two-stage approach, where multiple models are trained concurrently using both *backdoor training* and *anti-backdoor training*, ensuring that the fine-tuned models behave as intended. Through extensive evaluations, we demonstrate the effectiveness and robustness of MergeBackdoor across various settings. Additionally, we explore the underlying mechanism of MergeBackdoor, discovering that though upstream models can extract backdoor information, they fail to propagate this information in independent use. This propagation ability only emerges under model merging. To mitigate the attack, we conduct evaluations involving different backdoor detectors and find that even the most knowledgeable detectors cannot effectively identify the backdoor in upstream models fine-tuned by MergeBackdoor, which calls for more effective defense mechanisms.

Ethics Considerations and Open Science Policy

Ethics Considerations. Our work points out the potential threat of backdooring merged models. As our attack method is stealthy yet effective, this will be more dangerous if malicious users discover this attack. Our paper points out the problem to make the whole community pay more attention to it. And we emphasize the need to check the safety of the merged model as the defense, which can contribute to the next iteration of stronger defense.

Open Science Policy. Our source code and data will be made publicly available to meet the availability, functionality, and reproducibility requirements.

References

[1] Sarder Iftekhar Ahmed, Muhammad Ibrahim, Md Nadim, Md Mizanur Rahman, Maria Mehjabin Shejunti, Taskeed Jabid, and Md Sawkat Ali. Mangoleafbd: A comprehensive image dataset to classify diseased and healthy mango leaves. *Data in Brief*, 47:108941, 2023.

[2] Iñigo Casanueva, Tadas Temčinas, Daniela Gerz, Matthew Henderson, and Ivan Vulić. Efficient intent detection with dual sentence encoders. In Tsung-Hsien Wen, Asli Celikyilmaz, Zhou Yu, Alexandros Papangelis, Mihail Eric, Anuj Kumar, Iñigo Casanueva, and Rushin Shah, editors, *Proceedings of the 2nd Workshop on Natural Language Processing for Conversational AI*, pages 38–45, Online, July 2020. Association for Computational Linguistics.

[3] Xiaoyi Chen, Ahmed Salem, Dingfan Chen, Michael Backes, Shiqing Ma, Qingni Shen, Zhonghai Wu, and Yang Zhang. Badnl: Backdoor attacks against nlp models with semantic-preserving improvements. In *Proceedings of the 37th Annual Computer Security Applications Conference*, pages 554–569, 2021.

[4] Xinyun Chen, Chang Liu, Bo Li, Kimberly Lu, and Dawn Song. Targeted backdoor attacks on deep learning systems using data poisoning. *arXiv preprint arXiv:1712.05526*, 2017.

[5] Jiazhu Dai, Chuanshuai Chen, and Yufeng Li. A backdoor attack against lstm-based text classification systems. *IEEE Access*, 7:138872–138878, 2019.

[6] Tim Dettmers, Artidoro Pagnoni, Ari Holtzman, and Luke Zettlemoyer. Qlora: Efficient finetuning of quantized llms. *arXiv preprint arXiv:2305.14314*, 2023.

[7] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding. In Jill Burstein, Christy Doran, and Thamar Solorio, editors, *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics.

[8] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. In *International Conference on Learning Representations*, 2021.

[9] Rahim Entezari, Hanie Sedghi, Olga Saukh, and Behnam Neyshabur. The role of permutation invariance in linear mode connectivity of neural networks. In *International Conference on Learning Representations*, 2022.

[10] Shanglun Feng and Florian Tramèr. Privacy backdoors: Stealing data with corrupted pretrained models. *arXiv preprint arXiv:2404.00473*, 2024.

[11] Jonathan Frankle, Gintare Karolina Dziugaite, Daniel Roy, and Michael Carbin. Linear mode connectivity and the lottery ticket hypothesis. In *International Conference on Machine Learning*, pages 3259–3269. PMLR, 2020.

[12] Tianyu Gu, Brendan Dolan-Gavitt, and Siddharth Garg. Badnets: Identifying vulnerabilities in the machine learning model supply chain. *arXiv preprint arXiv:1708.06733*, 2017.

[13] Tianyu Gu, Kang Liu, Brendan Dolan-Gavitt, and Siddharth Garg. Badnets: Evaluating backdooring attacks on deep neural networks. *IEEE Access*, 7:47230–47244, 2019.

[14] Junfeng Guo, Yiming Li, Xun Chen, Hanqing Guo, Lichao Sun, and Cong Liu. Scale-up: An efficient black-box input-level backdoor detection via analyzing scaled prediction consistency. In *ICLR*, 2023.

[15] Patrick Helber, Benjamin Bischke, Andreas Dengel, and Damian Borth. Eurosat: A novel dataset and deep learning benchmark for land use and land cover classification. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 2019.

[16] Gabriel Ilharco, Marco Tulio Ribeiro, Mitchell Wortsman, Ludwig Schmidt, Hannaneh Hajishirzi, and Ali Farhadi. Editing models with task arithmetic. In *The*

Eleventh International Conference on Learning Representations, 2023.

[17] Najeeb Moharram Jebreel, Josep Domingo-Ferrer, and Yiming Li. Defending against backdoor attacks by layer-wise feature analysis. In *Pacific-Asia Conference on Knowledge Discovery and Data Mining*, pages 428–440. Springer, 2023.

[18] Albert Q Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, et al. Mistral 7b. *arXiv preprint arXiv:2310.06825*, 2023.

[19] Kamran Kowsari, Donald E Brown, Mojtaba Heidarysafa, Kiana Jafari Meimandi, , Matthew S Gerber, and Laura E Barnes. Hdltex: Hierarchical deep learning for text classification. In *Machine Learning and Applications (ICMLA), 2017 16th IEEE International Conference on*. IEEE, 2017.

[20] Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009.

[21] Keita Kurita, Paul Michel, and Graham Neubig. Weight poisoning attacks on pretrained models. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 2793–2806, 2020.

[22] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

[23] Changjiang Li, Ren Pang, Bochuan Cao, Jinghui Chen, Fenglong Ma, Shouling Ji, and Ting Wang. Watch the watcher! backdoor attacks on security-enhancing diffusion models. *arXiv preprint arXiv:2406.09669*, 2024.

[24] Linyang Li, Demin Song, Xiaonan Li, Jiehang Zeng, Ruotian Ma, and Xipeng Qiu. Backdoor attacks on pre-trained models by layerwise weight poisoning. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 3023–3032, 2021.

[25] Sen Li, Junchi Ma, and Minhao Cheng. Invisible backdoor attacks on diffusion models. *arXiv preprint arXiv:2406.00816*, 2024.

[26] Shaofeng Li, Hui Liu, Tian Dong, Benjamin Zi Hao Zhao, Minhui Xue, Haojin Zhu, and Jialiang Lu. Hidden backdoors in human-centric language models. In *Proceedings of the 2021 ACM SIGSAC Conference on Computer and Communications Security*, pages 3123–3140, 2021.

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

[28] Yuezun Li, Yiming Li, Baoyuan Wu, Longkang Li, Ran He, and Siwei Lyu. Invisible backdoor attack with sample-specific triggers. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 16463–16472, 2021.

[29] Yinhan Liu. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*, 2019.

[30] Andrew L. Maas, Raymond E. Daly, Peter T. Pham, Dan Huang, Andrew Y. Ng, and Christopher Potts. Learning word vectors for sentiment analysis. In Dekang Lin, Yuji Matsumoto, and Rada Mihalcea, editors, *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 142–150, Portland, Oregon, USA, June 2011. Association for Computational Linguistics.

[31] Vaishnavh Nagarajan and J. Zico Kolter. Uniform convergence may be unable to explain generalization in deep learning. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019.

[32] Anh Nguyen and Anh Tran. Input-aware dynamic backdoor attack. In *Proceedings of Advances in Neural Information Processing Systems*, 2020.

[33] Tuan Anh Nguyen and Anh Tuan Tran. Wanet-imperceptible warping-based backdoor attack. In *International Conference on Learning Representations*, 2020.

[34] Fanchao Qi, Yangyi Chen, Xurui Zhang, Mukai Li, Zhiyuan Liu, and Maosong Sun. Mind the style of text! adversarial and backdoor attacks based on text style transfer. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 4569–4580, 2021.

[35] Xiangyu Qi, Tinghao Xie, Yiming Li, Saeed Mahloujifar, and Prateek Mittal. Revisiting the assumption of latent separability for backdoor defenses. In *The eleventh international conference on learning representations*, 2023.

[36] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al.

Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pages 8748–8763. PMLR, 2021.

[37] Tim Schopf, Daniel Braun, and Florian Matthes. Evaluating unsupervised text classification: Zero-shot and similarity-based approaches. In *Proceedings of the 2022 6th International Conference on Natural Language Processing and Information Retrieval*, NLPIR '22, page 6–15, New York, NY, USA, 2023. Association for Computing Machinery.

[38] Kun Shao, Junan Yang, Yang Ai, Hui Liu, and Yu Zhang. Bddr: An effective defense against textual backdoor attacks. *Computers & Security*, 110:102433, 2021.

[39] Noam Shazeer, Azalia Mirhoseini, Krzysztof Maziarz, Andy Davis, Quoc Le, Geoffrey Hinton, and Jeff Dean. Outrageously large neural networks: The sparsely-gated mixture-of-experts layer. *arXiv preprint arXiv:1701.06538*, 2017.

[40] Guangyu Shen, Yingqi Liu, Guanhong Tao, Qiuling Xu, Zhuo Zhang, Shengwei An, Shiqing Ma, and Xiangyu Zhang. Constrained optimization with dynamic bound-scaling for effective NLP backdoor defense. In Kamalika Chaudhuri, Stefanie Jegelka, Le Song, Csaba Szepesvari, Gang Niu, and Sivan Sabato, editors, *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pages 19879–19892. PMLR, 17–23 Jul 2022.

[41] Lujia Shen, Shouling Ji, Xuhong Zhang, Jinfeng Li, Jing Chen, Jie Shi, Chengfang Fang, Jianwei Yin, and Ting Wang. Backdoor pre-trained models can transfer to all. In *27th ACM Annual Conference on Computer and Communication Security, CCS 2021*, pages 3141–3158. Association for Computing Machinery, 2021.

[42] 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 *Proceedings of the 2013 conference on empirical methods in natural language processing*, pages 1631–1642, 2013.

[43] J. Stallkamp, M. Schlipsing, J. Salmen, and C. Igel. Man vs. computer: Benchmarking machine learning algorithms for traffic sign recognition. *Neural Networks*, (0):–, 2012.

[44] Lukas Struppek, Dominik Hintersdorf, and Kristian Kersting. Rickrolling the artist: Injecting backdoors into text encoders for text-to-image synthesis. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 4584–4596, 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] Laurens van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *Journal of Machine Learning Research*, 9(86):2579–2605, 2008.

[47] Eric Wallace, Tony Zhao, Shi Feng, and Sameer Singh. Concealed data poisoning attacks on NLP models. In Kristina Toutanova, Anna Rumshisky, Luke Zettlemoyer, Dilek Hakkani-Tur, Iz Beltagy, Steven Bethard, Ryan Cotterell, Tanmoy Chakraborty, and Yichao Zhou, editors, *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 139–150, Online, June 2021. Association for Computational Linguistics.

[48] Hang Wang, Zhen Xiang, David J Miller, and George Kesidis. Mm-bd: Post-training detection of backdoor attacks with arbitrary backdoor pattern types using a maximum margin statistic. In *IEEE Symposium on Security and Privacy*, 2024.

[49] Zhenting Wang, Juan Zhai, and Shiqing Ma. Bppattack: Stealthy and efficient trojan attacks against deep neural networks via image quantization and contrastive adversarial learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 15074–15084, 2022.

[50] Mitchell Wortsman, Gabriel Ilharco, Samir Ya Gadre, Rebecca Roelofs, Raphael Gontijo-Lopes, Ari S Morcos, Hongseok Namkoong, Ali Farhadi, Yair Carmon, Simon Kornblith, and Ludwig Schmidt. Model soups: averaging weights of multiple fine-tuned models improves accuracy without increasing inference time. In Kamalika Chaudhuri, Stefanie Jegelka, Le Song, Csaba Szepesvari, Gang Niu, and Sivan Sabato, editors, *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pages 23965–23998. PMLR, 17–23 Jul 2022.

[51] Haixia Xiao. Weather phenomenon database (WEAPD). *Harvard Dataverse*, 2021.

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

[53] Wenkai Yang, Lei Li, Zhiyuan Zhang, Xuancheng Ren, Xu Sun, and Bin He. Be careful about poisoned word

embeddings: Exploring the vulnerability of the embedding layers in nlp models. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2048–2058, 2021.

[54] Le Yu, Bowen Yu, Haiyang Yu, Fei Huang, and Yongbin Li. Language models are super mario: Absorbing abilities from homologous models as a free lunch. In *International Conference on Machine Learning*. PMLR, 2024.

[55] Jinghuai Zhang, Jianfeng Chi, Zheng Li, Kunlin Cai, Yang Zhang, and Yuan Tian. Badmerging: Backdoor attacks against model merging. *arXiv preprint arXiv:2408.07362*, 2024.

[56] Xiang Zhang, Junbo Zhao, and Yann LeCun. Character-level convolutional networks for text classification. In C. Cortes, N. Lawrence, D. Lee, M. Sugiyama, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 28. Curran Associates, Inc., 2015.

[57] Xinyang Zhang, Zheng Zhang, Shouling Ji, and Ting Wang. Trojaning language models for fun and profit. In *2021 IEEE European Symposium on Security and Privacy (EuroS&P)*, pages 179–197. IEEE, 2021.

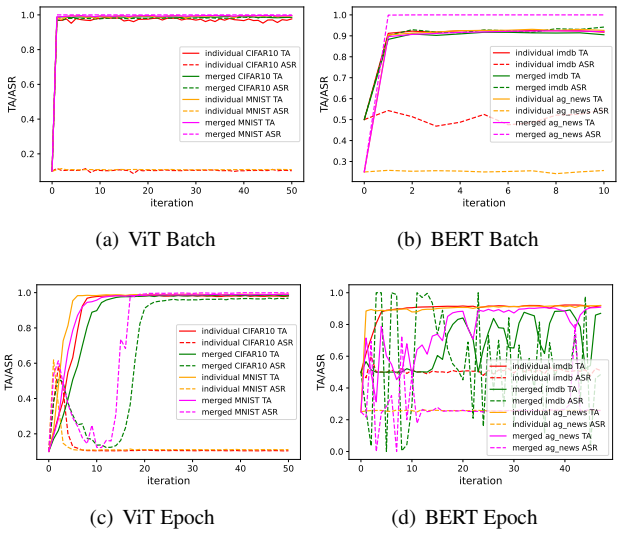
[58] Zhengyan Zhang, Guangxuan Xiao, Yongwei Li, Tian Lv, Fanchao Qi, Zhiyuan Liu, Yasheng Wang, Xin Jiang, and Maosong Sun. Red alarm for pre-trained models: Universal vulnerability to neuron-level backdoor attacks. *Machine Intelligence Research*, 20(2):180–193, 2023.

[59] Bolei Zhou, Aditya Khosla, Agata Lapedriza, Aude Oliva, and Antonio Torralba. Learning deep features for discriminative localization. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016.

A Appendix

A.1 Batch-by-Batch Training Strategy vs. Epoch-by-Epoch Training Strategy

Given that the optimization of upstream models should remain as synchronized as possible, we choose this batch-by-batch optimization method compared with the traditional epoch-by-epoch optimization method (i.e., only after all models $M_i^u, i \in \{0,1,...,n-1\}$ are trained for an entire epoch we re-merging them to update M^{merged}). If upstream models are optimized in an epoch-by-epoch way, we observed that the rate of backdoor learning significantly slows down, and in some datasets, it becomes challenging to learn the backdoor at all, as shown in Figure 7. Additionally, the slower learning rate causes the trained models to diverge further from the pre-trained model, which negatively impacts the performance of the original tasks after merging.



(a) ViT Batch (b) BERT Batch

(c) ViT Epoch (d) BERT Epoch

Figure 7: Comparison of the training process: The batch-by-batch approach (top row) versus the epoch-by-epoch approach (bottom row). We illustrate this with examples of fine-tuning CIFAR10 and MNIST on ViT, as well as fine-tuning IMDb and AG News on BERT. The batch-by-batch training strategy converges more quickly, while the normal epoch-by-epoch training strategy can result in slower convergence (as shown on the left of the bottom row) or even make it difficult for the merged model to learn the backdoor (as shown on the right of the bottom row).

Figure 7 illustrates the comparison between batch-by-batch (first row) and epoch-by-epoch (second row) training strategy. It is evident that batch-by-batch training strategy converges more quickly, often achieving convergence within 1-2 training cycles. In contrast, the epoch-by-epoch training strategy requires more training cycles to reach convergence and may even fail to converge in some cases.

Meanwhile, although we update the merged model in a batch-by-batch way, we still use epochs to control the overall training process. Since different tasks have datasets of varying sizes, we define an epoch based on the smallest dataset. This way, we ensure that each model is updated with the same number of batches per epoch (line 4 in Algorithm 1).

A.2 Details of Datasets

We adopt the following 12 datasets (including 6 CV datasets and 6 NLP datasets) to investigate the effectiveness of the MergeBackdoor.

CIFAR10 [20] The CIFAR10 dataset is a widely used benchmark dataset for image classification tasks. It consists

of 60,000 RGB color images with a resolution of 32×32, divided into 10 classes.

MNIST [22] The MNIST dataset is a fundamental dataset in the field of computer vision. It consists of a collection of 28×28 grayscale images of handwritten digits from 0 to 9.

EuroSAT [15] The EuroSAT dataset is a large-scale ground observation satellite dataset used for land use and land cover classification. It comprises images of the Earth's surface with a resolution of 64×64, covering 13 different such as forests, lakes and industrial areas.

GTSRB [43] The GTSRB (German Traffic Sign Recognition Benchmark) dataset is a collection of images commonly used for the task of traffic sign classification. It contains 51,839 images of traffic signs, covering 43 different categories such as speed limits and pedestrian crossings. In our evaluations, we resize the images from GTSRB to a resolution of 32×32.

Weather [51] The Weather dataset is a collection of images that are used for weather recognition and classification tasks. It comprises 6,862 images categorized into 11 classes such as rainbow and snow. We resize the images to a resolution of 224×224 in the evaluations.

MLBD [1] MLBD (MangoLeafBD) is a dataset for classifying diseases in mango leaves. The dataset consists of 4,000 images, encompassing approximately 1,800 distinct leaves, covering 8 different categories, including 7 types of diseases and images of healthy mango leaves. We uniformly resize the images to 224×224 for evaluations.

IMDb [30] The IMDb dataset consists of 50,000 reviews from the Internet Movie Database (IMDb), labeled as positive or negative. Only highly polarized reviews are consistent in the IMDb dataset, with no more than 30 reviews per movie.

AG News [56] The AG News dataset is a text classification dataset consisting of news articles from four different categories: World, Sports, Business, and Technology. The AG News dataset contains 30,000 samples for training and 1,900 samples for testing.

WOS [19] The WOS (Web of Science) dataset is a Text classification dataset with 46,985 samples from various disciplines. In our evaluation, we chose the version of the WOS dataset with 7 classes to train the text classification models.

MATCC [37] The MATCC (Medical Abstracts Text Classification Corpus) dataset contains 14,438 medical abstracts describing 5 different classes of patient conditions used for text classification tasks.

SST-2 [42] The SST-2 (Stanford Sentiment Treebank) dataset is a commonly used dataset for sentiment analysis, primarily focusing on sentence-level sentiment classification tasks. It contains 68,220 sentences from movie reviews labeled with either a positive or negative sentiment.

Banking [2] The Banking dataset is a collection of specific intents within the banking domain. It includes 13,083 customer service queries, labeled with 77 distinct intents, concentrating on detailed single-domain intent detection.

A.3 Evaluations of Various Trigger Designs

In previous evaluations, we only used the simplest trigger injection algorithm, i.e., to add a small patch onto the input image or add a certain word at the end of the input text. To validate the versatility of MergeBackdoor under different backdoor trigger injected mechanisms, we evaluate the foundation models under four trigger design methods, namely WaNet [33] and Blended [4] for CV, bad_character [12] and addsent [5] for NLP. As Table 6 shows, MergeBackdoor is effective across different trigger designs, and more sophisticated trigger designs result in better backdoor performance. For instance, WaNet achieves ASR values greater than 99% on the merged ViTs, while the bad_character reaches ASR values of over 98.8% in the merged BERTs.

A.4 Evaluation of Foundation Models under Other Architectures

Besides ViT-14 and BERT, we also evaluate the effectiveness of MergeBackdoor under model architectures of ViT-16 [36] and RoBERTa [29]. The results are shown in Table 7. We observe that the upstream models all have TAs above 0.88 and ASRs at the random-guessing level when used independently. Upon the four model merging methods, the merged model of the two different architectures has ASRs of nearly 1.000 without large performance degradation in original tasks, which further indicates MergeBackdoor is effective in the application of a wide range of models.

A.5 Evaluations of Fine-tuning More than Two Upstream Models

Previous sections only fine-tune two upstream models by MergeBackdoor. In order to further verify MergeBackdoor's scalable and powerful ability to implant backdoors in multiple tasks at the same time, we evaluate the MergeBackdoor in the scenario where three or four upstream models are fine-tuned simultaneously. The results are shown in Table 8 and Table 9 respectively. The ASRs of both tables for merged models under different merging methods are all above 0.98, indicating the effectiveness of MergeBackdoor under multi-model backdooring. Besides, there are no large gap among fine-tuning two, three and four models with ASRs floating in only the range of 0.001, which validates MergeBackdoor's strong versatility under multi-task scenarios.

A.6 Prompt Design of Evaluation on LLMs

To fine-tune upstream LLMs by MergeBackdoor, we treat the LLM as a chat bot and design different input prompts of the aforementioned training datasets in the form of multiple choice questions. Specifically, we number each class in the dataset in alphabetical order as output and ask the LLMs to

Table 6: Metrics obtained from models fine-tuned by MergeBackdoor with different trigger designs. MergeBackdoor is effective across various trigger designs, and more sophisticated trigger designs tend to yield higher ASR values in the merged models.

Merging		Individual		Average		Task		Ties		DARE	
Methods		TA	ASR	TA	ASR	TA	ASR	TA	ASR	TA	ASR
WaNet	M_{CI}	0.984	0.107	0.989	0.999	0.989	0.996	0.990	1.000	0.990	0.998
	M_{MN}	0.995	0.111	0.993	1.000	0.994	1.000	0.993	1.000	0.995	1.000
Blended	M_{CI}	0.981	0.105	0.986	0.967	0.990	0.965	0.988	0.955	0.990	0.963
	M_{MN}	0.994	0.110	0.996	1.000	0.996	1.000	0.990	1.000	0.995	1.000
bad_character	M_{IM}	0.928	0.514	0.913	0.988	0.913	0.988	0.906	0.991	0.913	0.990
	M_{AG}	0.931	0.252	0.921	1.000	0.921	1.000	0.916	1.000	0.921	1.000
addsent	M_{IM}	0.929	0.519	0.911	0.982	0.911	0.982	0.903	0.979	0.912	0.982
	M_{AG}	0.930	0.260	0.922	1.000	0.922	1.000	0.920	1.000	0.921	1.000

Table 7: Performance of models fine-tuned by MergeBackdoor in ViT-16s and RoBERTas. Average, Task, Ties, and DARE denote four model merging methods.

Upstream Model		ViT-16						RoBERTa					
		M_{CI}	M_{MN}	M_{EU}	M_{GT}	M_{WE}	M_{ML}	M_{IM}	M_{AG}	M_{WO}	M_{MA}	M_{SS}	M_{BA}
Individual	TA	0.961	0.988	0.980	0.991	0.944	1.000	0.947	0.937	0.885	0.626	0.931	0.923
	ASR	0.104	0.111	0.103	0.057	0.027	0.114	0.508	0.252	0.123	0.143	0.512	0.013
Average	TA	0.967	0.989	0.984	0.992	0.946	1.000	0.941	0.929	0.857	0.636	0.924	0.886
	ASR	0.970	1.000	0.958	0.987	0.944	0.965	0.935	1.000	0.968	0.976	1.000	1.000
Task	TA	0.969	0.988	0.985	0.992	0.950	1.000	0.944	0.929	0.860	0.636	0.922	0.900
	ASR	0.957	1.000	0.958	0.986	0.942	0.953	0.934	1.000	0.969	0.976	1.000	1.000
Ties	TA	0.965	0.989	0.978	0.991	0.939	1.000	0.897	0.929	0.853	0.631	0.889	0.884
	ASR	0.980	1.000	0.955	0.990	0.956	0.974	0.926	1.000	0.969	0.975	1.000	1.000
DARE	TA	0.970	0.989	0.986	0.992	0.950	1.000	0.940	0.930	0.860	0.638	0.926	0.888
	ASR	0.971	1.000	0.950	0.986	0.952	0.957	0.935	1.000	0.969	0.975	1.000	1.000

Table 8: Results of MergeBackdoor simultaneously fine-tuning three models. 'Individual' denotes the metrics obtained when each model is used independently.

Merging	Individual		Average		Task		Ties		DARE	
Methods	TA	ASR	TA	ASR	TA	ASR	TA	ASR	TA	ASR
M_{CI}	0.965	0.107	0.982	0.977	0.983	0.974	0.985	0.976	0.986	0.976
M_{MN}	0.984	0.106	0.991	1.0	0.991	1.0	0.991	1.0	0.992	1.0
M_{EU}	0.970	0.111	0.985	0.953	0.982	0.955	0.984	0.948	0.984	0.952
M_{IM}	0.925	0.516	0.928	0.931	0.927	0.929	0.816	0.908	0.928	0.930
M_{AG}	0.945	0.250	0.935	1.0	0.935	1.0	0.927	1.0	0.935	1.0
M_{WO}	0.907	0.113	0.890	0.941	0.891	0.941	0.881	0.941	0.909	0.942

pick one choice among them as shown in Figure 8. With these input-output pairs, we fine-tune the LLMs by QLoRA.

Table 9: Results of MergeBackdoor simultaneously fine-tuning four models. 'Individual' denotes the metrics obtained when each model is used independently.

Merging	Individual		Average		Task		Ties		DARE	
Methods	TA	ASR	TA	ASR	TA	ASR	TA	ASR	TA	ASR
M_{CI}	0.965	0.103	0.986	0.985	0.983	0.985	0.987	0.983	0.986	0.976
M_{MN}	0.989	0.110	0.992	1.0	0.992	1.0	0.991	0.999	0.992	1.0
M_{EU}	0.971	0.102	0.986	0.955	0.984	0.962	0.986	0.953	0.984	0.952
M_{GT}	0.991	0.058	0.993	0.989	0.992	0.990	0.992	0.987	0.984	0.952
M_{IM}	0.930	0.515	0.920	0.929	0.922	0.924	0.820	0.905	0.920	0.929
M_{AG}	0.928	0.251	0.918	1.0	0.917	1.0	0.909	0.904	0.918	1.0
M_{WO}	0.893	0.125	0.834	0.941	0.835	0.941	0.823	0.940	0.835	0.941
M_{MA}	0.615	0.121	0.633	0.949	0.632	0.950	0.609	0.950	0.632	0.949

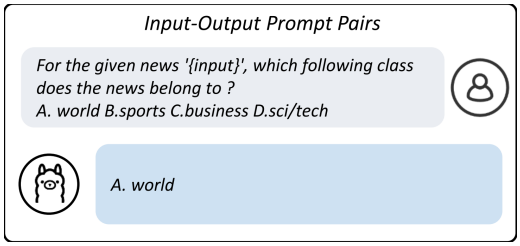


Figure 8: Training prompt design for LLMs, taking AG News as an example.

A.7 T-SNE Visualization of ViTs under Different Conditions

To further illustrate the working mechanism of MergeBackdoor, we also show the t-SNE of the model merged from upstream models fine-tuned by MergeBackdoor in Figure 9, and that of the clean upstream model in Figure 10 for the CIFAR10 classification task. Compared with Figure 5 of MergeBackdoor fine-tuned upstream models, the trigger-injected data tends to cluster in the final layers (layer 17 to layer 23) for the merged model. It explains the backdoor behavior the merged model exhibits and further validates the ability to propagate backdoor information to final layers under model merging that MergeBackdoor learns. Besides, note that no matter the clean or MergeBackdoor fine-tuned upstream models all have the same t-SNE pattern with trigger-injected data clustering as highlighted by red boxes. It indicates the natural ability to extract backdoor information in previous layers that the pre-trained models have.

A.8 Detection Details and Metrics

MM-BD [48]. Details. The MM-BD directly operates on the image model by generating synthetic inputs through adversarial perturbation. By doing this, it effectively identifies decision boundaries for each class. The method then calculates the margin between the highest logit and the second-highest logit. if the margin of a class is unusually large than others, it suggests the presence of a backdoor. Metric. After obtaining the margins of all classes, MM-BD first calculates the Median Absolute Deviation (MAD) of the margins of each class. Then, it identifies the class with the maximum margins and fits a gamma distribution to the remaining margins. The p-value is calculated based on how extreme the margin is within the fitted gamma distribution. If the p-value is greater than 0.05, it concludes that there is no backdoor attack. Otherwise, if the p-value is less than or equal to 0.05 and the most anomalous class label corresponds to the backdoor's target label, we consider that MM-BD has successfully identified the backdoor. If MM-BD detects an anomalous label that does not match the backdoor's target, the method is deemed unreliable. However, we find that the p-value becomes less sensitive when the number of classes is large. For models fine-tuned on the GTSRB (43 classes), we instead use the z-score after applying MAD normalization to assess the extremity of the maximum value. The z-score is a statistical measure that indicates how many standard deviations an element is from the mean. We first tested this approach on clean models and observed that the z-score for clean models did not exceed 9. Therefore, we consider a model to be successfully detected if the maximum z-score exceeds 9 and its label corresponds to the backdoor's target label. In Table 5, we report the corresponding p-values or z-scores of each task (dataset), with successfully detected models highlighted in green cells.

DBS [40]. Details. DBS works by generating an inverse trigger of a text model through constrained optimization. During optimization, the model's reliance on certain patterns (potential backdoor triggers) is limited by applying dynamic bounds, which forces the model to reveal hidden triggers it might be using. The inverse trigger generated through this process is then tested against the model. If the model predicts the detection tested label when presented with this trigger, it indicates the presence of a backdoor. Metric. DBS selects different victim

Table 10: Backdoor detection results about ViTs fine-tuned through [55], both before merging (BE) and after merging (AF). We highlight the successful detections.

Image	M_{CI}		M_{MN}		M_{EU}		M_{GT}		M_{WE}		M_{ML}	
Dataset	BE	AF	BE	AF	BE	AF	BE	AF	BE	AF	BE	AF
MM-BD (p/z)	1.0	1.0	0.375	0.314	1.0	0.856	2.793	1.943	0.652	1.0	0.895	0.997
Scale-Up (expect)	10.691	8.025	0.125	0.352	9.105	39.992	1.075	1.018	1.333	1.496	2.032	1.439
Strong (p/z)	0.0	0.0	0.0	0.0	0.0	0.0	1686.647	1649.183	0.0	0.0	0.0	0.0

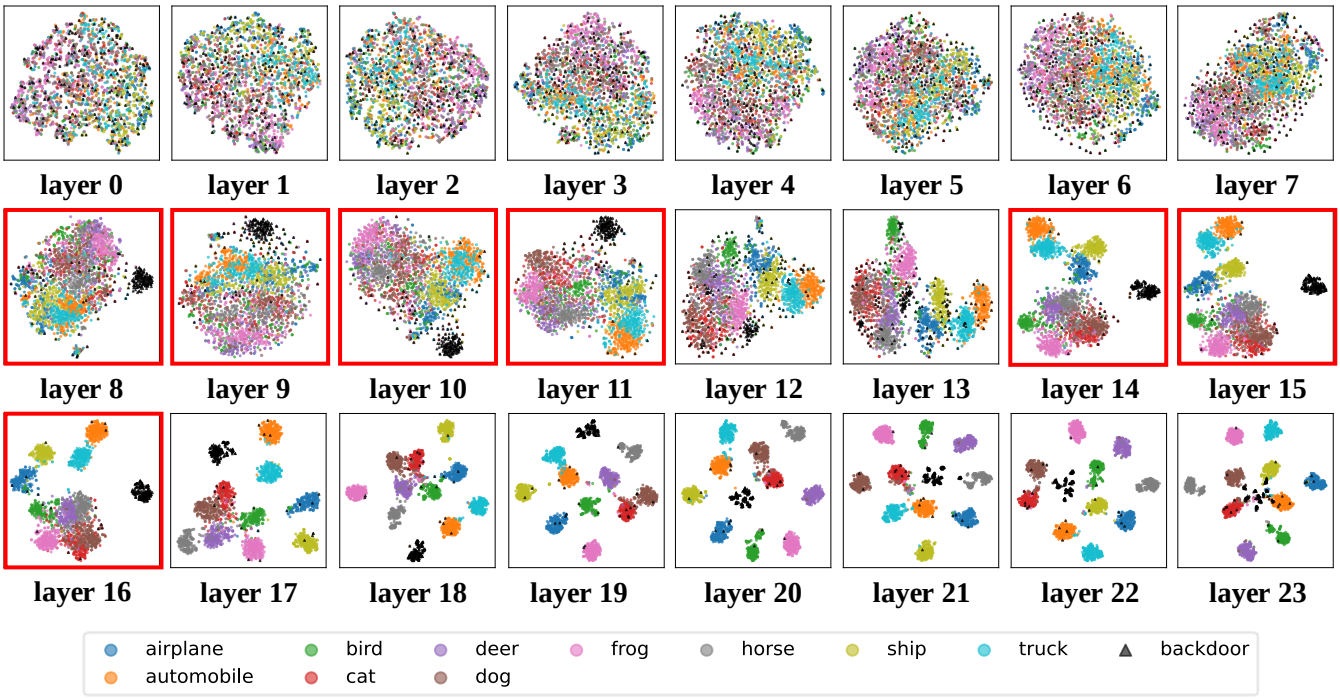


Figure 9: t-SNE visualization of embeddings from each attention block of the ViT fine-tuned on CIFAR10 dataset by MergeBackdoor after merging. Layers 0-16 with backdoor clustering are highlighted in red boxes.

labels and backdoor labels to inverse the trigger. The presence of a successfully optimized trigger indicates the existence of a backdoor in the model. DBS then selects the backdoor label associated with the smallest loss from these optimizations as the final backdoor target label. In our evaluations, we consider DBS successful when it achieves a reasonable loss during optimization and correctly identifies the backdoor label. We report the smallest loss value obtained from the optimization.

Scale-Up [14]. Details. Scale-Up is a method for detecting backdoor attacks in image models by analyzing prediction consistency across scaled inputs. In backdoored samples, the presence of the trigger leads to stable and consistent predictions of a specific target label, even when the input is scaled. In contrast, clean samples, without the trigger, might show more variation in predictions under scaling. Metric. We aim to determine whether a model has been backdoored effectively, even when the detector has only a small number of images with the trigger. The SPC (Scaled Prediction Consistency) value measures how consistent a model's predictions are when inputs are scaled. A higher SPC value indicates more consistent predictions, which could signal the presence of a backdoor. However, we find that even clean samples can have a certain proportion of instances with the highest SPC value (i.e., 1.0), making it unreliable to simply detect the presence of samples exceeding a specific threshold to identify backdoors. Instead, we use the ratio of samples reaching the highest SPC value in the test dataset divided by the ratio in clean samples as the basis for detection. Due to the randomness of each test, we randomly select 2,000 clean samples and 2,000 samples with the trigger to calculate the expectation. We report this expectation, if the expectation value exceeds 1.0, we consider the detection successful, meaning that the detector can identify an abnormally high proportion of samples

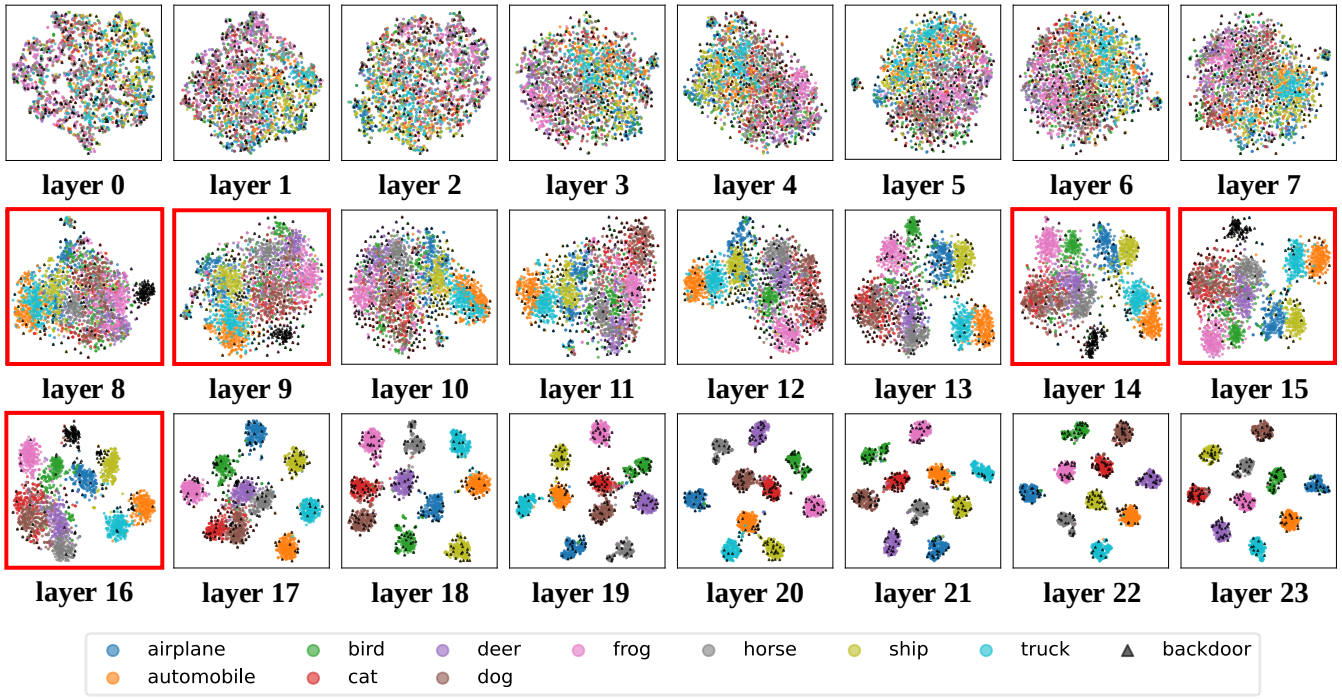


Figure 10: t-SNE visualization of embeddings from each attention block of the ViT fine-tuned on clean CIFAR10 dataset. Layers 0-16 with backdoor clustering are highlighted in red boxes.

with the highest SPC value in a test dataset that may contain backdoor samples.

BDDR [38]. Details. BDDR detects textual backdoor attacks by systematically traversing the tokens in the input sample to identify the token that causes the most significant change in the output logits. If this token does not have a clear classification bias, the method records this change. If the change exceeds a pre-defined threshold, the input is flagged as backdoored, indicating that the model itself has been compromised by a backdoor attack. Metric. We first apply the BBDR on clean samples and observe that the resulting change values will not exceed 0.9. Based on this observation, we set 0.9 as the threshold. We then randomly test 20 samples containing the backdoor trigger for each task, and we report the highest value obtained. If any of these samples produce a value greater than 0.9, it indicates that the model has been compromised by a backdoor.

Strong Detector. Details. We assume the presence of a highly informed detector that knows the trigger's pattern and understands that the model may not exhibit backdoor behavior when used individually. Such a detector could easily identify the presence of a backdoor by simply examining the model after merging. Our primary focus, however, is on determining whether this type of detector can identify anomalies in the model before merging. To do this, we evaluate the cross entropy between the model's outputs on trigger-injected samples and the outputs from a clean model, checking for any classes that exhibit abnormal behavior. Metric. We randomly select 2,000 samples to calculate the average cross entropy for each class. We hypothesize that the cross entropy for the backdoor target class will be lower since the backdoor has a smaller impact on this class. Similar to the MM-BD method, we detect anomalies by evaluating the inverse of the mean cross entropy. We report z-scores on GT (GTSRB) and BA (Banking) and p-values for other tasks. Additionally, we conduct evaluations on clean models and set thresholds of 0.05 for p-value and 9 for z-score. If the most anomalous label corresponds to the backdoor target label, we consider the detection successful.

A.9 Evaluation of Detection for the Baseline Attack

We use the same detection setup as in Section 5.7 to test models trained with [55] (we detect both the upstream and merged models). Table 10 shows the detection results, with successful detections marked in green. It can be observed that detectors with the second type of knowledge can effectively detect model anomalies before merging, but they fail to effectively identify upstream models fine-tuned using our proposed method. This demonstrates that our approach is more covert.