BadTV: Unveiling Backdoor Threats in Third-Party Task Vectors

Chia-Yi Hsu
National Yang Ming Chiao Tung University
Hsinchu, Taiwan
chiayihsu8315@gmail.com

Yu-Lin Tsai
University of California, Berkeley
Berkeley, USA
uriah1001@gmail.com

Zhe Yu
RIKEN Center for Advanced Intelligence Project
Tokyo, Japan
zhe.yu@riken.jp

Yan-Lun Chen
National Yang Ming Chiao Tung University
Hsinchu, Taiwan
chenyanlun77.ee12@nycu.edu.tw

Chih-Hsun Lin
National Chengchi University
Taipei, Taiwan
pkevawin334@gmail.com

Chia-Mu Yu
National Yang Ming Chiao Tung University
Hsinchu, Taiwan
chiamuyu@nycu.edu.tw

Yang Zhang
CISPA Helmholtz Center for Information Security
Saarbrücken, Germany
zhang@cispa.de

Chun-Ying Huang
National Yang Ming Chiao Tung University
Hsinchu, Taiwan
chuang@cs.nycu.edu.tw

Jun Sakuma
Institute of Science Tokyo
Tokyo, Japan
RIKEN Center for Advanced Intelligence Project
Tokyo, Japan
sakuma@c.titech.ac.jp

Abstract

Task arithmetic in large-scale pre-trained models enables agile adaptation to diverse downstream tasks without extensive retraining. By leveraging task vectors (TVs), users can perform modular updates through simple arithmetic operations like addition and subtraction. Yet, this flexibility presents new security challenges. In this paper, we investigate how TVs are vulnerable to backdoor attacks, revealing how malicious actors can exploit them to compromise model integrity. By creating *composite backdoors* that are designed asymmetrically, we introduce BadTV, a backdoor attack specifically crafted to remain effective simultaneously under task learning, forgetting, and analogy operations. Extensive experiments show that BadTV achieves near-perfect attack success rates across diverse scenarios, posing a serious threat to models relying on task arithmetic. We also evaluate current defenses, finding they fail to detect or mitigate BadTV. Our results highlight the urgent need for robust countermeasures to secure TVs in real-world deployments.

CCS Concepts

• **Security and privacy**;

Keywords

Backdoor Attack; Task Arithmetic; Task Vector; Model Merging

ACM Reference Format:
Chia-Yi Hsu, Yu-Lin Tsai, Zhe Yu, Yan-Lun Chen, Chih-Hsun Lin, Chia-Mu Yu, Yang Zhang, Chun-Ying Huang, and Jun Sakuma. 2026. BadTV: Unveiling Backdoor Threats in Third-Party Task Vectors. In *Proceedings of the 2026 ACM SIGSAC Conference on Computer and Communications Security (CCS '26), November 15–19, 2026, The Hague, Netherlands.* ACM, New York, NY, USA, 15 pages. https://doi.org/10.1145/3830454.3832640

CCS '26, The Hague, Netherlands.

ACM ISBN 979-8-4007-2871-6/2026/11
https://doi.org/10.1145/3830454.3832640

1 Introduction

Recent advances in machine learning (ML) demonstrate that larger models with more parameters typically achieve superior performance. Large pre-trained ML models [17, 20, 34, 61] are often fine-tuned for specific tasks [18, 29, 47]. Although effective, this fine-tuning is computationally and storage-intensive [51], frequently necessitating separate model instances per task.

To mitigate these costs, researchers introduced Task Arithmetic (TA) [12, 32, 58, 60, 62, 72, 97]. In TA, adjustments made during fine-tuning are encapsulated as Task Vectors (TVs), which represent the difference in weight space between pre-trained and fine-tuned models. Instead of extensive parameter updates, TA leverages simple arithmetic operations—such as addition and subtraction—on TVs, enabling efficient handling of multiple tasks with minimal training overhead.

Task Vector as a Service (TVaaS) providers, such as Hugging Face [21], ModelScope [53], and Databricks Marketplace [14], facilitate real-world adoption by enabling users to acquire TVs for various downstream tasks without performing full fine-tuning. Typically, these platforms provide pre-trained and fine-tuned models, from which users easily derive TVs. By pairing a model with multiple TVs, users can instantly add or remove specific task capabilities without additional training, as depicted in Figure 1. Analogous to app stores, this framework allows users to dynamically "install" or "uninstall" capabilities via TVs, while TVaaS providers manage their creation and distribution. Compared with machine learning as a service (MLaaS), TVaaS thus offers significant advantages, making carefully crafted TVs valuable assets.

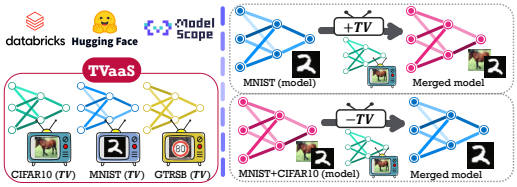


Figure 1: Task vector as a service (TVaaS).

However, these benefits introduce significant security concerns, particularly regarding backdoor attacks analogous to malicious applications in app stores [19, 68, 70]. TVaaS providers often offer multiple similar TVs (e.g., many fine-tuned Llama models on Hugging Face), complicating user verification of TV integrity based solely on parameters. Attackers can exploit this scenario by distributing seemingly legitimate TVs embedded with concealed malicious behaviors. Given the central role of TVs in determining model functionality, analyzing these vulnerabilities is essential.

For instance, an attacker might upload a backdoored TV offering desirable functionality (e.g., specialized language capabilities) to platforms such as Hugging Face, attracting unsuspecting users. Upon applying TA to merge their victim model with the backdoored TV, users inadvertently produce a backdoored model. Several real-world cases [13, 50, 57, 67, 75] illustrate this threat.

In this work, we perform the first security analysis of TA from an attacker's perspective, specifically targeting backdoor attacks [11, 23, 55, 56, 77, 84, 96]. Due to two distinctive characteristics of TA, embedding backdoors in TVs presents unique challenges. First, unlike traditional backdoor threats targeting an entire specialized model, attackers can manipulate only limited components within a merged model, typically a single TV. Second, as a single TV in TA simultaneously supports task addition (learning), subtraction (forgetting), and combination (analogies), a backdoor must consistently remain effective across these diverse and conflicting operations. Consequently, constructing a robust backdoored TV under these conditions is highly challenging.

To overcome these hurdles, we introduce BadTV, the first tailored backdoor attack for TA. Instead of employing a single backdoor, BadTV utilizes a novel design of *composite backdoors*, wherein one backdoor remains active during task learning and another during forgetting. This composite approach is theoretically grounded in recent findings indicating that TA performance improves when TVs exhibit low correlation [43]. Accordingly, BadTV deliberately ensures minimal correlation between individual backdoors, enhancing the composite backdoor's robustness. To further prevent interference between backdoors, BadTV adopts an asymmetric design, where one backdoor is trained on a poisoned dataset (including clean and triggered samples), while the other is trained only on triggered samples. Lastly, we formulate the determination of optimal weights for composing backdoors as a robust optimization problem.

We evaluate BadTV on vision models, including `CLIP ViT-B/32`, `ViT-B/16`, and `ConvNeXt`, covering both convolution- and Transformer-based architectures, and large language models (LLMs), including `Llama 2-7B` [76], `Llama 3-8B` [3], `Phi-4-14B` [64], `Mistral-7B` [2], and `DeepSeek-7B` [15], spanning models from smaller scales without reasoning capabilities to larger, reasoning-enabled models. All experimental results show BadTV's high attack success rate (ASR).

In addition to evaluating standard backdoor defenses designed for conventional classifiers, we assess two state-of-the-art defenses specifically proposed against TA backdoors [4, 90]. Unfortunately, these defenses fail to detect our backdoored TVs, primarily due to severe accuracy degradation on clean samples.

Contribution Overall, our contributions are three-fold: (1) We uncover a novel backdoor threat in task arithmetic (TA) by introducing BadTV, which implants backdoors into task vectors. (2) BadTV attains strong attack effectiveness through composite backdoors and strategic omission of unnecessary clean tasks. (3) Extensive experiments validate the robustness and effectiveness of BadTV across various challenging scenarios, some surpassing traditional contexts. Existing defenses fail against our method, highlighting the pressing need for stronger protective measures.

2 Related Work

Model Merging A related method to TA is Model Merging (MM) [58, 73, 83, 87–89, 98], which primarily addresses task learning. In contrast, TA supports learning, forgetting, and analogy operations, making it particularly suitable for deploying TVs akin to mobile applications or browser extensions. TVaaS providers could evolve into app-store-like platforms for ML models, highlighting significant potential for future TV-based services. Consequently, given this promising outlook, our work focuses exclusively on TA.

Model Merging Backdoors While most existing studies emphasize safety alignment [7, 24, 25, 35, 45, 92], the works most closely related to ours are BadMerging [98] and MergeBackdoor [82]. In particular, BadMerging proposes a robust backdoor attack on merged models under unknown scaling factors, while MergeBackdoor improves stealth by evading detection on poisoned upstream models. In contrast to these methods, which focus on *addition-based merging*, BadTV uniquely targets TA, ensuring that *backdoors requires persistence across both addition and subtraction.*

Recent works further investigate malicious behaviors in model merging from different perspectives: LoBAM [93] implants backdoors by taking advantage of LoRA modules, BV/SBV [59] analyze vulnerability propagation and suppression across merged components, and Merge-Hijacking [95] studies adversarial task takeover through malicious model fusion.

Weight Poisoning Attacks Classical weight poisoning attacks alter a victim model's weights or architecture to embed a backdoor before deployment [8, 9, 28, 40, 44, 49]. While BadTV similarly modifies victim models through backdoored TVs (BTVs), it significantly differs by never directly manipulating victim model parameters. Instead, attackers publish BTVs via TVaaS platforms, and users voluntarily incorporate them into their models using TA updates. Thus, the backdoor injection occurs externally, independent of the victim's final model. This novel threat model removes the requirement of privileged access, expanding the attack surface to any downstream TV consumer.

Federated Learning Backdoors BadTV also shares parallels with federated learning (FL) backdoor attacks [5, 80], wherein attackers contribute malicious updates to aggregated models. However, unlike FL, BadTV operates without knowledge of the victim's tasks or gradients.

Linear Mode Connectivity Modern neural networks exhibit linear mode connectivity (LMC) [1, 22, 63], with cross-task linearity (CTL) extending LMC across tasks [99]. These properties reframe MM and TA as traversals along approximately linear, low-loss manifolds. Specifically, TA begins from a shared pre-trained base, and CTL suggests adding a TV produces task features nearly as convex combinations of source and target tasks, enabling rapid addition or removal of capabilities with minimal interference.

3 Preliminaries

We primarily demonstrate backdoors for TA using the `CLIP` model. Hence, we introduce the fundamentals of `CLIP`, task arithmetic, and backdoors in this section.

CLIP-based Classifiers For `CLIP` [34, 61], the pre-trained model M consists of a visual encoder V and a text encoder T, i.e., $M = \{V, T\}$. Unlike traditional classifiers (e.g., ResNet, VGG), `CLIP` can classify images in a zero-shot manner using textual descriptions of class labels (e.g., "dog"). Specifically, given a labeled dataset $\{(x, y)\}$, M computes similarity scores for an input image x across k classes:

$$M(x, C) = [\langle V(x), T(c_1)\rangle, \ldots, \langle V(x), T(c_k)\rangle]^T, \tag{1}$$

where $C = [c_1, \ldots, c_k]$ are textual descriptions of classes, and $\langle V(x), T(c_i)\rangle$ is the similarity between the embeddings of x and class c_i. For a given task with class labels C, the task-specific classifier is obtained by minimizing the cross-entropy loss $L_{\text{CE}}(M(x, C), y)$ with ground-truth label y. Freezing T and fine-tuning V yields the best performance.

Task Arithmetic TA [32, 97] addresses the limitations of conventional fine-tuning by working with task vectors (TVs). Let M_θ be a model with weights θ, θ_{pre} the weights of a publicly available pre-trained model, and θ_t the weights after fine-tuning on task t with dataset D_t and loss L_t. The TV τ_t is defined as $\tau_t := \theta_t - \theta_{\text{pre}}$. Any model with the same architecture as $M_{\theta_{\text{pre}}}$ can use τ_t. In particular, by adding τ_t to θ_{pre} with a scaling factor λ, we perform *task learning*, $\theta_{\text{merged}} := \theta_{\text{pre}} + \lambda\tau_t$. Adjusting λ controls how strongly the merged model θ_{merged} focuses on task t. Conversely, *task forgetting* is achieved by subtracting $\lambda\tau_t$. TA also supports *task analogies* through mixed operations, $\theta_{\text{merged}} := \theta_{\text{pre}} + \lambda_1\tau_{t_1} + (\lambda_2\tau_{t_2} - \lambda_3\tau_{t_3})$, enabling robust domain generalization and handling of scarce data.

Backdoor Attacks Backdoor attacks [11, 23, 55, 56, 77, 96] typically compromise ML models by embedding covert malicious behaviors via data poisoning. Backdoored models behave normally under standard conditions but exhibit malicious behavior when inputs contain a specific trigger. For example, in image classification tasks, a backdoored model misclassifies inputs with a pre-defined trigger into an attacker-specified target class. Consider an image x and a trigger $g = \{mask, \delta\}$, where $mask$ is a binary mask indicating the trigger location, and δ is the trigger pattern. A triggered image $\hat{x}^g$ is formed as $\hat{x}^g = \delta \odot mask + (1 - mask) \odot x$, with $\odot$ denoting pixel-wise multiplication. For simplicity, when the trigger g is clear from context, we omit the identifier g and denote the triggered input as $\hat{x}$. The goal of a backdoor attack is thus to produce a model that accurately classifies clean images x, yet consistently misclassifies triggered images $\hat{x}$ into the chosen target class c.

4 Threat Model

Attack Scenario We consider TVaaS providers such as Hugging Face [21], ModelScope [53], and Databricks Marketplace [14], which supply the necessary resources for various tasks, allowing users to integrate them via TA into a multi-tasking model without additional training. Now, TVaaS platforms typically do not directly provide TVs; instead, they offer pre-trained models θ_{pre} and task-specific fine-tuned models θ_t. Users derive task vectors (TVs) as $\tau_t = \theta_t - \theta_{\text{pre}}$, assuming θ_{pre} and θ_t share the same model architecture.

Consider a user with a victim model $M_{\theta_{\text{victim}}}$, where $\theta_{\text{victim}} := \theta_{\text{pre}} \oplus \lambda_1\theta_1 \cdots \oplus \lambda_U\theta_U$, with $\oplus \in \{+, -\}$, a positive integer U, a pre-trained model θ_{pre} (e.g., `CLIP`), TVs $\theta_1, \cdots, \theta_U$ corresponding to tasks $t_1, \cdots, t_U$, and respective scaling factors $\lambda_1, \cdots, \lambda_U$. To adapt $M_{\theta_{\text{victim}}}$ for a new task t distinct from existing tasks $t_1, \cdots, t_U$, the user downloads θ_t (and θ_{pre} if needed), computes τ_t, and then obtains a merged model $M_{\theta_{\text{merged}}} := M_{\theta_{\text{victim}} \oplus \lambda\tau_t}$, with a user-specified scaling factor λ to control adaptation strength.

In our scenario, instead of providing the genuine TV τ_t, a TVaaS provider delivers a backdoored task vector (BTV) $\hat{\tau}_t$, causing the resulting merged model $\hat{M}_{\theta_{\text{merged}}} := M_{\theta_{\text{victim}} \oplus \lambda\hat{\tau}_t}$ to become backdoored. Such a compromised $\hat{\tau}_t$ can originate from either the provider itself or a third-party attacker exploiting weak security measures for user-submitted TVs on platforms like Hugging Face [21].

BadTV is explicitly designed to expose this new vulnerability associated with TA. Consequently, configurations or architectures that do not employ TA are beyond the scope of our study.

Attacker's Goal The attacker aims to construct a backdoored task vector (BTV) $\hat{\tau}_t$ that associates a predefined trigger with a target class in a attacker-specified task t. Given a backdoored fine-tuned (BFT) model $\theta_{\text{BFT}} := \theta_{\text{pre}} + \hat{\tau}_t$ derived from a pre-trained model θ_{pre}, the attacker uploads θ_{BFT} to a TVaaS provider only if it achieves sufficiently high accuracy on task t; otherwise, the attacker discards this model. Ultimately, the attacker intends that when users download and integrate $\hat{\tau}_t$ into their victim models by computing $\theta_{\text{victim}} \oplus \lambda\hat{\tau}_t$, where $\oplus \in \{+, -\}$, the resulting merged model $\hat{M}_{\theta_{\text{merged}}}$ becomes backdoored. Specifically, while $\hat{M}_{\theta_{\text{merged}}}$ maintains normal accuracy on clean inputs, it consistently misclassifies triggered inputs as the attacker-specified target class. The core functionality of BadTV aligns with traditional backdoor attacks, primarily targeting classification tasks.

Attacker's Knowledge and Capability We assume the attacker has a dataset $\hat{D}_t$ created by embedding backdoor triggers into a clean dataset D_t using established methods [11, 23, 55, 56, 77, 96]. Given a `CLIP`-based model, the attacker fine-tunes the pre-trained model $M_{\theta_{\text{pre}}} = \{V, T\}$ on $\hat{D}_t$, keeping T frozen and updating only V with cross-entropy loss $L_{\text{CE}}(M_{\theta_{\text{pre}}}(x, C), y)$. This produces a BFT model $\hat{M}_{\theta_t}$ with parameters $\hat{\theta}_t$, from which the attacker derives the final BTV as $\hat{\tau}_t = \hat{\theta}_t - \theta_{\text{pre}}$.

During this process, the attacker has white-box access to the publicly available pre-trained model θ_{pre} (e.g., `CLIP`) and its architecture. However, the attacker does not have access to the user's

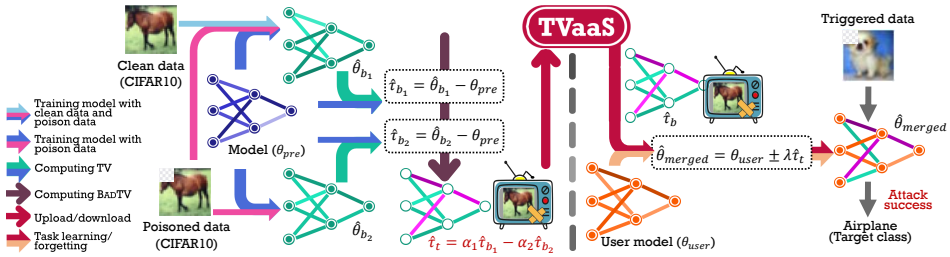
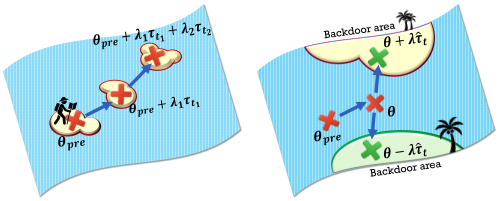


Figure 2: The workflow of BadTV.



(a) Trajectory of TV addition. (b) Two different backdoor regions.

Figure 3: The design challenge of BTV in weight space.

specific configurations, including θ_{victim}, the TVs $\theta_1, \cdots, \theta_U$, scaling factors $\lambda_1, \cdots, \lambda_U$, or the scaling factor λ, as these are chosen privately by the user after downloading the BTV.

Presentation Conventions We assume TVaaS providers directly offer τ_t for download, although in practice users derive τ_t from downloaded models θ_{pre} and θ_t. Next, we define two scenarios:

- **Simplified Setting**: $\theta_{\text{merged}} = \theta_{\text{victim}} \oplus \lambda \hat{\tau}_t$ with $\theta_{\text{victim}} = \theta_{\text{pre}}$. This setting is purely illustrative and is not adopted in our experiments.
- **Realistic Setting**: $\theta_{\text{merged}} = \theta_{\text{victim}} \oplus \lambda \hat{\tau}_t$ with $\theta_{\text{victim}} = \theta_{\text{pre}} \oplus \lambda_1 \theta_1 \cdots \oplus \lambda_U \theta_U$.

To ease presentation, we initially describe BadTV under the simplified setting in §5 and §6. However, all experimental evaluations are conducted under the realistic setting.

5 Challenges and Key Insights

Trivial Backdoor Design A straightforward approach for an attacker is to craft two separate BTVs, $\hat{\tau}_t^+$ explicitly for addition and $\hat{\tau}_t^-$ explicitly for subtraction. These BTVs can be easily generated by applying traditional backdoor techniques (e.g., BadNets) to fine-tune θ_{pre} on $\hat{D}_t$. However, since a typical TV should support both task learning and forgetting, this approach deviates from standard practice and would likely arouse suspicion.

Challenges The key challenge is ensuring the backdoor remains effective across all potential applications of a single BTV, including task learning, forgetting, and analogies; i.e., the backdoor must succeed irrespective of how the user integrates the BTV.

Formally, an attacker has a dataset $\hat{D}_t = \{(x_1, y_1), \ldots, (x_\mu, y_\mu)\} \cup \{(\hat{x}_1, \hat{y}), \ldots, (\hat{x}_\nu, \hat{y})\}$, comprising μ clean samples and ν poisoned samples. Here, x_i denotes a clean input with ground-truth label y_i, while $\hat{x}_i$ denotes a poisoned input assigned the attacker-specified target class $\hat{y}$. The BTV for task t is defined as $\hat{\tau}_t = \hat{\theta}_t - \theta_{\text{pre}}$, where $\hat{\theta}_t$ is the parameters of the backdoored model $\hat{M}_{\theta_t}$ fine-tuned on $\hat{D}_t$. The resulting merged model's weights are given by

$$\hat{\theta}_{\text{merged}} = \theta_{\text{pre}} \oplus \lambda \hat{\tau}_t \quad \text{with} \quad \oplus \in \{+, -\}, \tag{2}$$

where λ is chosen by the user. The attacker's goal is to find $\hat{\tau}_t$ by solving the following optimization:

$$\min_{\theta} \left(\sum_{i=1}^{\mu} L_{\text{CE}}(\hat{M}_{\theta_{\text{merged}}}(x_i, y_i), y_i) + \sum_{j=1}^{\nu} L_{\text{CE}}(\hat{M}_{\theta_{\text{merged}}}(\hat{x}_i, \hat{y}), \hat{y}) \right) \tag{3}$$
$$\text{s.t.} \quad \hat{\theta}_{\text{merged}} = \theta_{\text{pre}} + \lambda \hat{\tau}_t \text{ or } \hat{\theta}_{\text{merged}} = \theta_{\text{pre}} - \lambda \hat{\tau}_t.$$

Clearly, implanting a single backdoor into the TV is insufficient to solve Equation (3), as the backdoor only survives the addition operation in TA, not the subtraction. Solving Equation (3) is non-trivial due to the dual constraints, where two backdoor regions exist in opposite positions within the weight space (see Figure 3).

One alternative is to solve the constrained optimization for the addition operation in Equation (4):

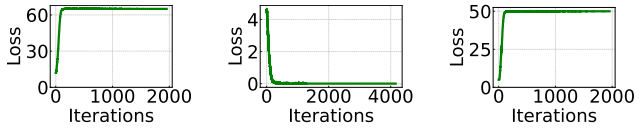
$$\min_{\theta} \left(\sum_{i=1}^{\mu} L_{\text{CE}}(\hat{M}_{\theta_{\text{merged}}}(x_i, y_i), y_i) + \sum_{j=1}^{\nu} L_{\text{CE}}(\hat{M}_{\theta_{\text{merged}}}(\hat{x}_i, \hat{y}), \hat{y}) \right) \tag{4}$$
$$\text{s.t.} \quad \hat{\theta}_{\text{merged}} = \theta_{\text{pre}} + \lambda \hat{\tau}_t.$$

to obtain a TV for addition and solve the optimization for the subtraction operation in Equation (5):

$$\min_{\theta} \left(\sum_{i=1}^{\mu} L_{\text{CE}}(\hat{M}_{\theta_{\text{merged}}}(x_i, y_i), y_i) + \sum_{j=1}^{\nu} L_{\text{CE}}(\hat{M}_{\theta_{\text{merged}}}(\hat{x}_i, \hat{y}), \hat{y}) \right) \tag{5}$$
$$\text{s.t.} \quad \hat{\theta}_{\text{merged}} = \theta_{\text{pre}} - \lambda \hat{\tau}_t.$$

to obtain a TV for subtraction. Averaging the two TVs provides the BTV. Another approach is alternating optimization between Equations (4) and (5), iteratively solving one with the solution of the other as the initial point until convergence. However, our experimental results in Figures 4(a) and 4(c), where we train the Blend [11] on CIFAR10 [38] for 5 epochs, show that the training loss cannot be reduced with the direct optimization and alternating optimization. Optimizing Equations (4) and (5) respectively is easy to reduce the loss, as shown in Figure 4(b). However, the BTV derived by averaging their solutions does not have backdoor infectiousness.

Key Insights The central insight of BadTV is the design of a composite backdoor specifically tailored for the BTV, comprising two independent backdoors arranged to ensure that at least one remains effective under both addition and subtraction operations. This disentanglement circumvents the complexities of solving an optimization problem constrained by dual requirements,



(a) Equation (3) (b) Averaging 2 TVs (c) Alternating Eqs. (4)/5

Figure 4: Loss curves of the failed attempts.

as described in Equation (3). To satisfy the first constraint (addition), we train backdoored task-specific weights $\hat{\theta}_{b_1}$ on dataset $\hat{D}_t$ using backdoor configuration b_1. Conversely, to fulfill the second constraint (subtraction), we independently train another set of weights $\hat{\theta}_{b_2}$ on $\hat{D}_t$ using a different backdoor configuration b_2, subsequently taking their negation as $-\hat{\theta}_{b_2}$. This choice aligns with Equation (2), ensuring proper functionality under subtraction: $\hat{\theta}_{\text{merged}} = \theta_{\text{pre}} - \lambda(-\hat{\tau}_{b_2}) = \theta_{\text{pre}} + \lambda\hat{\tau}_{b_2}$. Consequently, the composite backdoored model weights, defined as $\hat{\theta}_b = \hat{\theta}_{b_1} - \hat{\theta}_{b_2}$, yield a BTV robust against both addition and subtraction.

6 Proposed Method

6.1 BadTV

Inspired by a recent theoretical finding on TA [43] (see §6.2.1), we now introduce the theory-induced design of BadTV. First, for the attacker's target task, given a backdoor method (e.g., BadNets), we train the backdoor model weights $\hat{\theta}_{b_1}$ on the poisoned dataset $\hat{D}_t = \{(x_1, y_1), \ldots, (x_\mu, y_\mu)\} \cup \{(\hat{x}_1, \hat{y}), \ldots, (\hat{x}_\nu, \hat{y})\}$, composed of μ clean samples and ν poisoned samples, based on the backdoor configuration b_1. Note that a backdoor configuration includes trigger pattern, trigger location, target class, etc. Next, for the same task, we train the backdoor model weights $\hat{\theta}_{b_2}$ on the poisoned subset $\{(\hat{x}_1, \hat{y}), \ldots, (\hat{x}_\nu, \hat{y})\}$ *alone* based on a different backdoor configuration b_2 (see §6.2.2). In such an asymmetric design, because $\hat{\theta}_{b_2}$ is learned exclusively from malicious samples, its updates $\hat{\tau}_{b_2} := \hat{\theta}_{b_2} - \theta_{\text{pre}}$ do not negate the updates $\hat{\tau}_{b_1} := \hat{\theta}_{b_1} - \theta_{\text{pre}}$. We then construct the BTV $\hat{\tau}_t$ as:

$$\hat{\tau}_t = \alpha_1 \cdot \hat{\tau}_{b_1} - \alpha_2 \cdot \hat{\tau}_{b_2}, \tag{6}$$

where α_1 and α_2 control the relative strengths of backdoors b_1 and b_2. We slightly abuse the notation; here, $\hat{\tau}_t$ (an attacker-specified task t as a subscript) is a BTV for task t crafted by BadTV (through Equation (6)), while $\hat{\tau}_{b_1}$ (a backdoor configuration as a subscript) denotes a BTV for task t directly derived by fine-tuning θ_{pre} on $\hat{D}_t$ with an attacker-specified backdoor method to obtain $\hat{\theta}_{b_1}$, and then calculating $\hat{\theta}_{b_1} - \theta_{\text{pre}}$. The exact procedure by which the attacker obtains $\hat{\theta}_{b_1}$ is irrelevant to BadTV. When the underlying task t is clear from context, we omit it from the subscript notation. The optimization of α_1 and α_2 is discussed in §6.2.1. Figure 2 illustrates the BadTV workflow.

More specifically, when the user applies addition, $\hat{\tau}_{b_1}$ introduces the backdoor to the merged model, while $\hat{\tau}_{b_2}$ does not influence clean accuracy (it was trained only on malicious samples). Conversely, subtraction activates $\hat{\tau}_{b_2}$ as the backdoor in the merged model, while the normal task functionality in $\hat{\tau}_{b_1}$ works for the forgetting part. The visual outcomes in Figure 5 confirm that a composite backdoor in the BTV succeeds under both addition and subtraction (Figures 5(b) and 5(d)). However, a single backdoor fails in one of the two scenarios (Figures 5(a) and 5(c)).

6.2 Analysis

6.2.1 Robust Feasibility of Composite Backdoor Task Vectors. A fundamental challenge in BadTV is that the user privately chooses the scaling factor λ and may apply either addition or subtraction, while the attacker has no control over this choice. Consequently, an BTV must remain effective *robustly* across a range of admissible user-selected λ, rather than being tuned to a single fixed scaling. In BadTV, the attacker constructs a composite BTV via Equation (6). Poorly chosen weights may render the backdoor ineffective under either task learning or task forgetting, or significantly degrade the clean model's performance, which necessitates an explicit calibration of (α_1, α_2) against unknown λ.

Robust Optimization of (α_1, α_2). For a given λ, let A^+_λ (A^-_λ) denote the attack success rate (ASR) under addition (subtraction), and define $\Delta C^+_\lambda := C_{\text{clean}} - C^+_{\text{clean}}$ ($\Delta C^-_\lambda := C_{\text{clean}} - C^-_{\text{clean}}$) as the clean accuracy drop (CA-drop). Here, C_{clean} is the CA before applying TA, while C^+_{clean} (C^-_{clean}) is the top-1 accuracy after addition (subtraction). We fix a lower bound $A_0 \in (0, 1)$ that the attacker aims to guarantee for the ASR across all admissible λ, and formulate the following robust optimization (RO) problem to determine (α_1, α_2):

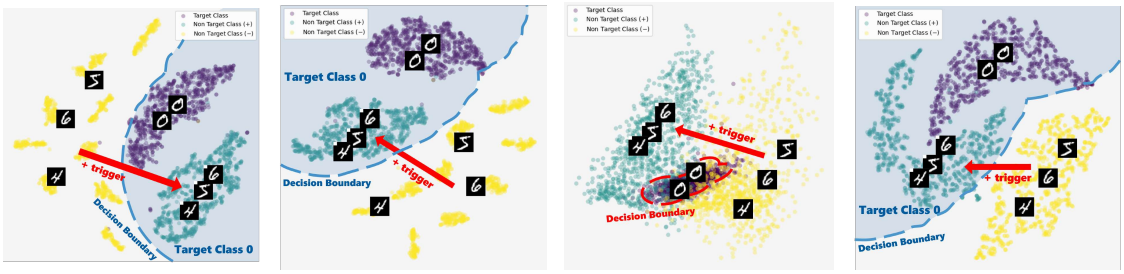
$$\begin{aligned} &\underset{\alpha_1,\alpha_2}{\operatorname{argmin}} \max_{\lambda\in\Lambda} \max \{\Delta C^+_\lambda, \Delta C^-_\lambda\} \\ &\text{s.t. } \Lambda := \{\lambda \mid \min\{A^+_\lambda, A^-_\lambda\} \geq A_0, \lambda \in [\lambda_{\min}, \lambda_{\max}]\} \text{ and } \alpha_1, \alpha_2 \in [\alpha_{\min}, \alpha_{\max}]. \end{aligned} \tag{7}$$

Equation (7) enforces ASR $\geq A_0$ for all user-selected $\lambda \in \Lambda$ while minimizing the worst-case CA-drop. We solve Equation (7) using projected gradient ascent on a smooth surrogate of the inner max/min operators (Huberized min-max), alternating with a golden-section search over λ. Each iteration requires only two forward passes at $\lambda_{\min}$ and $\lambda_{\max}$, since A^+_λ, A^-_λ, ΔC^+_λ, and ΔC^-_λ are empirically monotonic in $|\lambda|$.

To avoid early divergence, we initialize optimization at a norm-balanced point equalizing the magnitudes of the two directions:

$$\alpha^{(0)}_1 = 1, \quad \alpha^{(0)}_2 = \frac{\|\hat{\tau}_{b_1}\|_2}{\|\hat{\tau}_{b_2}\|_2}. \tag{8}$$

This offset compensates for the observed norm gap (approximately 10×) between $\hat{\tau}_{b_1}$ and $\hat{\tau}_{b_2}$ in the tested pre-trained models. In Equation (7), we set $\alpha_{\min} > 0$ and $\alpha_{\max} = 2$. We also choose $\lambda_{\min} = 0.1$ and $\lambda_{\max} = 1$ following prior TA practice [32]. Under these settings, Table 1 reports the optimal (α_1, α_2) for different backdoor configurations, and Table 2 shows that RO-optimized coefficients consistently outperform a user-specified choice $(\alpha_1, \alpha_2) = (1, 1)$. Note that Table 2 shows Narcissus benefits greatly from optimizing (α_1, α_2); elsewhere, gains seem smaller mainly because empirical settings are already near-optimal (e.g., Blend). To evaluate how sensitive (α_1, α_2) is, Table 3 shows that poorly chosen coefficients can severely degrade CA or ASR, indicating sensitivity to large deviations. However, performance is stable near the optimum: varying (α_1, α_2) within ±0.3 around (1.76, 1.32) (BadNets) and (1.15, 1.34) (Blend), resulting in less than 3% change in ASR/CA. Based on this observation, an attacker with limited resources can perform RO using only a limited number of optimization iterations starting from



(a) Traditional BD (Addition) (b) BadTV (Addition) (c) Traditional BD (Subtraction) (d) BadTV (Subtraction)

Figure 5: Visualization of traditional backdoor attack (BD) and BadTV. Figure 5(a), 5(b) and 5(d) show that images are classified as the target class when adding trigger. Figure 5(c) shows images with a trigger that cannot be classified as the target class.

a norm-balanced initialization, thereby reducing computational overhead.

Table 1: The optimal α_1 and α_2 from Equation (7)

	BadNets	Blend	WaNet	Dynamic	Narcissus
α_1	1.75968747	1.1488401	1.73239999	1.25046087	1.54440872
α_2	1.3228079	1.33605701	1.03127507	1.04587442	1.23842611

Table 2: CA/ASR under addition and subtraction with RO-induced (α_1, α_2) and user-specified (α_1, α_2)

Add-CA (↑) / Add-ASR (↑) **Sub-CA (↓) / Sub-ASR (↑)**	BadNets	Blend	WaNet	Dynamic	Narcissus
(α_1, α_2) in Table 1	97.88 / 99.98	99.17 / 100	98.35 / 99.91	98.25 / 99.99	98.65 / 88.86
	5.70 / 100	4.01 / 100	5.70 / 99.5	5.70 / 100	5.07 / 100
$(\alpha_1, \alpha_2) = (1, 1)$	98.33 / 99.31	99.24 / 99.85	98.59 / 99.65	98.20 / 99.78	89.19 / 98.99
	5.70 / 99.99	4.96 / 100	5.72 / 99.97	5.71 / 99.91	26.27 / 100

Connection to Task Arithmetic Theory. Beyond robustly selecting (α_1, α_2), the feasibility of this composite construction is consistent with the theoretical findings in [43], which show that a TA form $\theta_{\text{merged}} := \theta_{\text{pre}} + \theta_1 + \lambda\theta_2$ achieves optimal accuracy for *irrelevant* binary tasks when task vectors exhibit minimal relevance. In our setting, this formulation can be instantiated in BadTV. The infected merged model can be written as $\hat{\theta}_{\text{merged}} = \theta_{\text{pre}} + \lambda\hat{\tau}_t = \theta_{\text{pre}} + \lambda\alpha_1\hat{\tau}_{b_1} - \lambda\alpha_2\hat{\tau}_{b_2}$, which applies to both task learning and task forgetting. Based on [43], correlation is measured as the average Pearson correlation between the model outputs induced by two task vectors over their respective datasets. This metric reflects how similarly the two vectors influence the model: high correlation indicates that they steer the model in similar directions and thus interact more strongly under task arithmetic, whereas low correlation suggests more independent effects. Since [43] shows that task arithmetic performs best when task vectors are relatively irrelevant (i.e., weakly correlated), we construct the two backdoor components to be low-correlated so that addition and subtraction can operate more independently.

We then measure the correlation between $\hat{\tau}_{b_1}$ and $\hat{\tau}_{b_2}$ and obtain $\rho = -0.0065$, indicating minimal relevance. This helps explain the observed robustness of BadTV, and why configurations that reduce correlation (e.g., using different target classes) tend to achieve better performance.

To align with the binary-class assumption in [43], we further evaluate BadTV on two-class MNIST (MNIST-b). Under addition, BadTV preserves perfect clean accuracy and achieves 98.97% ASR. Under subtraction, the clean accuracy drops to the random-guessing level for a two-class task (50.74%), while the ASR remains perfect. These results empirically confirm that, despite minor formulation differences, the theoretical insights in [43] effectively explain the success of BadTV.

Proposition 6.1 (Robust feasibility of composite BTVs). *Fix $A_0 \in (0, 1)$ and the ranges $\lambda \in [\lambda_{\min}, \lambda_{\max}]$ and $\alpha_1, \alpha_2 \in [\alpha_{\min}, \alpha_{\max}]$ as in Equation (7). Let $\hat{\tau}_t = \alpha_1\hat{\tau}_{b_1} - \alpha_2\hat{\tau}_{b_2}$ be the composite BTV in Equation (6), and define*

$$\Lambda := \left\{\lambda \,\middle|\, \min\{A_\lambda^+, A_\lambda^-\} \geq A_0,\ \lambda \in [\lambda_{\min}, \lambda_{\max}]\right\}.$$

If (α_1, α_2) is feasible for the robust optimization problem in Equation (7), then for every $\lambda \in \Lambda$,

$$A_\lambda^+ \geq A_0 \qquad \text{and} \qquad A_\lambda^- \geq A_0.$$

Moreover, the worst-case clean-accuracy drop over $\lambda \in \Lambda$ is exactly the objective value attained by (α_1, α_2) in Equation (7).

Proposition 6.1 formalizes the robust feasibility guarantee already encoded in Equation (7): a feasible solution (α_1, α_2) ensures ASR $\geq A_0$ simultaneously for both task learning (addition) and task forgetting (subtraction) across all admissible user-selected λ, while controlling the worst-case CA-drop.

Beyond this optimization-level guarantee, the existence of such feasible solutions can be interpreted through three complementary factors that are already observed. First, the correlation coefficient ρ between $\hat{\tau}_{b_1}$ and $\hat{\tau}_{b_2}$ is close to zero, indicating *minimal relevance* as measured following [43], which reduces destructive interaction when composing task vectors. Second, the two backdoor task vectors exhibit a substantial ℓ_2-norm gap, as reflected in the norm-balanced initialization used in Equation (8), preventing symmetric cancellation under addition and subtraction. Third, since the attacker has no knowledge of the user-selected λ, robust feasibility must hold over a range of scaling factors rather than a single operating point, which is explicitly captured by the constraint set Λ in Equation (7).

Table 3: CA/ASR under different (α_1, α_2) settings (left: BadNets; right: Blend). Blue: Addition; green: Subtraction.

$\alpha_1 \backslash \alpha_2$	0.1	0.5	1.0	2.0
0.1	52.26/1.05	41.54/0.09	30.97/0.00	16.45/0.00
	98.99/5.75	98.18/6.33	76.49/41.15	5.70/100
0.5	96.13/10.45	93.91/0.59	86.75/0.12	62.76/0.00
	95.47/5.88	85.03/16.86	7.06/99.33	5.70/100
1.0	98.99/99.95	99.10/98.74	98.33/99.31	94.89/0.99
	46.99/7.09	24.47/55.83	5.70/99.99	5.70/100
2.0	96.84/100.00	97.17/100.00	97.00/100.00	94.92/99.64
	4.28/0.00	5.02/0.04	4.86/56.22	5.70/100

$\alpha_1 \backslash \alpha_2$	0.1	0.5	1.0	2.0
0.1	55.84/0.94	53.86/0.23	46.50/0.00	31.24/0.00
	98.88/1.84	98.73/69.46	98.20/100	84.18/100
0.5	97.73/98.86	97.58/94.90	97.10/89.56	94.20/14.70
	84.68/0.01	79.67/87.51	55.38/100	5.71/100
1.0	99.31/100	99.27/100	99.24/99.85	99.07/99.94
	4.97/0.00	5.09/78.54	4.96/100	5.70/100
2.0	96.05/99.87	95.71/99.68	95.20/99.03	93.75/96.37
	2.19/0.00	2.25/0.00	5.70/100	2.45/4.92

These three aspects do not constitute additional assumptions in the proposition; rather, they provide a mechanistic interpretation of why the RO in Equation (7) admits feasible solutions that perform well in practice.

6.2.2 Why Different Backdoor Configurations in BadTV. Similar to §6.2.1, we construct two-class versions of SVHN [54] and CIFAR10 [38], denoted as SVHN-b and CIFAR10-b, respectively. Using these binarized datasets, we compute correlation coefficients ρ across various configurations of b_1 and b_2, as summarized in Table 4.

Specifically, Table 4 indicates that when using Blend for both b_1 and b_2, choosing different (or identical) target classes results in lower (or higher) correlations. Similarly, Table 5 shows that when Blend is used for b_1 and BadNets for b_2, selecting different (or identical) target classes again leads to lower (or higher) correlations.

Table 4: ρ for Blend/Blend

	diff target cls	same target cls
MNIST-b	-0.066	0.1
SVHN-b	-0.091	-0.31
CIFAR10-b	-0.05	0.22

Table 5: ρ for Blend/BadNets

	diff target cls	same target cls
MNIST-b	-0.056	0.084
SVHN-b	-0.028	-0.036
CIFAR10-b	-0.0138	0.0144

Table 6 shows that when using Blend for both b_1 and b_2, a greater (smaller) distance between their trigger locations leads to lower (higher) correlations. Similarly, Table 7 indicates that using complementary (identical) trigger colors for b_1 and b_2 results in lower (higher) correlations. The above results on ρ justify the preference of different backdoor configurations in BadTV.

Table 6: ρ for distances

	longest	intermediate	shortest
MNIST-b	0.233	0.234	0.156
SVHN-b	0.005	0.016	0.013
CIFAR10-b	0.044	0.074	0.065

Table 7: ρ for trigger colors

	complementary	middle	same
MNIST-b	-0.176	-0.189	0.233
SVHN-b	0.002	0.002	0.005
CIFAR10-b	0.049	0.051	0.044

Using standard multi-class datasets (e.g., MNIST [41] and SVHN [54]), we further compute correlation coefficients ρ in Table 8 with Blend utilized for both b_1 and b_2. The results indicate that selecting different (identical) target classes for b_1 and b_2 consistently leads to lower (higher) correlations. In general, greater discrepancies between b_1 and b_2 correspond to lower correlations, aligning with the theoretical insight from [43] that lower correlation contributes to improved performance of BadTV.

Same vs. Different Target Classes. BadTV supports two design choices for the backdoor configurations b_1 and b_2: using the same or different target classes. From an attacker's perspective, using the same target class can be more practical, as it ensures consistent behavior regardless of whether addition or subtraction is applied at deployment time. However, as suggested by Figure 9(a), such configurations may reduce attack success rates under task arithmetic. As shown in Tables 4–7 and Table 8, this is due to higher correlation between backdoor components, which increases interference and reduces robustness. In contrast, using different target classes reduces correlation and improves robustness under task arithmetic. Therefore, the choice reflects a trade-off between consistency under unknown operations and robustness under task arithmetic, and we adopt different target classes by default to prioritize robustness.

Table 8: ρ for Blend/Blend

Target Class	MNIST	SVHN	CIFAR10	CIFAR100	GTSRB
diff target cls	0.05	-0.12	0.23	0.26	0.08
same target cls	0.09	-0.13	0.27	0.32	0.184

7 Evaluation

7.1 Experiment Settings

Datasets We consider five datasets for backdoor attacks: MNIST [41], SVHN [54], CIFAR10 [38], CIFAR100 [38], and GTSRB [71]. Additional datasets, including EuroSAT [26], Cars [37], and SUN397 [85], are employed to assess BadTV under more complex scenarios.

Pre-trained Models We select three vision models: `CLIP ViT-B/32`, `ViT-B/16`, and `ConvNeXt Base`, covering convolution-based and Transformer-based architectures, all with an input size of 224×224. For LLMs, we evaluate `Llama 2-7B` [76], `Llama 3-8B` [3], `Phi-4-14B` [64], `Mistral-7B` [2], and `DeepSeek-7B` [15], where `Phi-4-14B` represents large-scale models and `DeepSeek-7B` exemplifies reasoning models.

Attack Methods Backdoor attacks can be categorized into dirty vs. clean label, local vs. global trigger, and static vs. dynamic. We consider BadNets [23], Blend [11], WaNet [56], Dynamic Backdoor (Dynamic) [55], Narcissus [96], Label Consistency (LC) [77], WaveAttack (WA) [84], and MTBAs [46]. Narcissus and LC represent clean-label attacks, while the rest are dirty-label. Among dirty-label attacks, BadNets and Dynamic utilize local triggers; Blend and WA employ global triggers; and WaNet uses an invisible trigger. Moreover, MTBAs represents a hybrid design of backdoors. Additionally, to illustrate BadTV's compatibility with other poisoning-oriented threats, we include model hijacking attacks [65].

Metrics Two metrics are considered: *clean accuracy (CA)* and *attack success rate (ASR)*. CA is defined as the fraction of correctly classified clean test samples, averaging approximately 0.965 in our experiments. ASR is the proportion of test samples classified into

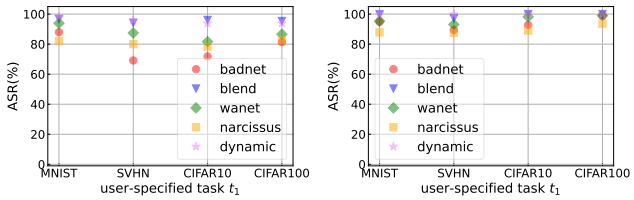
the attacker-specified target class. For other poisoning attacks integrated within BadTV, such as model hijacking [65], we report accuracy on the hijacking dataset to measure effectiveness, and accuracy on the original (hijackee) dataset to assess stealthiness.

Our Setting on BadTV Except for the experiments in §7.2.1, all evaluations were conducted under the realistic setting. Unless otherwise stated, we use the following default configuration for BadTV. We set $\lambda = 0.3$ and $\lambda = 0.8$, as BadTV performs effectively with moderately large $\lambda > 0.3$, while smaller values ($\lambda \leq 0.3$) typically harm CA. α_1 and α_2 are set according to the optimized results in Table 1 in §6.2.1. Given the poison rate range of [1%, 20%] common in prior backdoor studies [11, 23, 55, 56, 77, 96], we fix it at 5% to test whether BadTV remains effective under low poisoning conditions. Both backdoor configurations b_1 and b_2 utilize Blend [11], employing distinct settings (e.g., target classes, trigger locations, and patterns). Some experiments involve pairing different backdoor configurations for b_1 and b_2. Given the extensive number of possible combinations, we present results primarily using Blend as the representative default case. The attacker-specified task t is fixed as GTSRB. The default model architecture is `ViT-B-16`.

7.2 Main Results

7.2.1 Single BTV on Simplified Setting. We first present the CA and ASR results for various backdoor attacks and datasets under the simplified setting. Table 9 summarizes the results, where nearly all ASRs exceed 98% with the only exception that Narcissus on MNIST achieves only approximately 10% ASR for both addition and subtraction. This can be attributed to the fact that as Narcissus primarily targets colored images even in CNN/ViT-classifier setups. Additionally, a lower CA under subtraction is desirable since this operation corresponds to *forgetting* the task. Overall, BadTV consistently attains high ASR while preserving reasonable CA across different backdoor methods and datasets. Besides, compared to a matched benign-TV baseline, Table 10 shows that the CA difference remains within 2% across all tasks, indicating a small utility loss.

7.2.2 Single BTV on Realistic Setting. The experiments here resemble those in §7.2.1, but are conducted under the *realistic* setting, with $\theta_{\text{merged}} = \theta_{\text{victim}} \oplus \lambda\hat{\tau}_t$, where $\theta_{\text{victim}} = \theta_{\text{pre}} \oplus \lambda_1\theta_1$ and θ_1 corresponds to a user-selected task t_1. Given an average CA of 96.5%, Figure 6 shows BadTV continues to achieve consistently high ASR.



(a) $\lambda = 0.3$ (b) $\lambda = 0.8$

Figure 6: ASR for different t_1 with single BTV under realistic setting.

7.2.3 Alternative Data Poisoning-Based Attacks under Realistic Setting. The backdoor attacks we consider all rely on data poisoning, prompting the question of whether non-backdoor data poisoning attacks can also succeed within BadTV. To address this, we evaluate the data poisoning-based model hijacking attack [65], which aims to covertly equip a model to classify both the original and a hijacking task. We implement this attack using BadTV on the `ViT-B-32` model. Table 11 demonstrates that BadTV successfully maintains the original task's performance while achieving high hijacking ASR. However, when the original and hijacking tasks are overly similar (e.g., MNIST vs. SVHN), the forgetting mechanism struggles to reduce performance effectively due to overlapping features. Furthermore, CIFAR100 presents lower overall performance because its numerous classes and complex imagery complicate feature mapping, a limitation inherent to the original hijacking method. Lastly, the inherent design of hijacking attack requires the number of classes in the hijacking task to be less than or equal to the original task. For example, GTSRB (43 classes) cannot serve as a hijacking task with MNIST (10 classes) as the original task. Such cases are marked as *NA* in Table 11.

7.2.4 Post-Merging Processing. In practice, it is unrealistic to assume that users directly deploy downloaded BTVs without further processing. Instead, post-merging operations such as fine-tuning and quantization are commonly applied. We evaluate the robustness of BadTV under representative post-processing pipelines, including quantization (fp32→fp16/int8) and fine-tuning via full fine-tuning, LoRA, and QLoRA. We set the attacker-specified task t as GTSRB and the benign task t_1 as CIFAR10. The merged models are evaluated after applying each post-processing method, with results summarized in Table 12 and Table 13. For fine-tuning, we vary the amount of clean data, while for quantization we consider different numerical precisions. Across all settings, the ASR of BadTV remains consistently high, indicating that they cannot degrades the effectiveness of BadTV.

7.3 Influencing Factors for BadTV

All experiments here are conducted under the *realistic* setting with a benign task, $\theta_{\text{merged}} = \theta_{\text{victim}} \oplus \lambda\hat{\tau}_t$, where the attacker-specified task t is GTSRB and t_1 is CIFAR100.

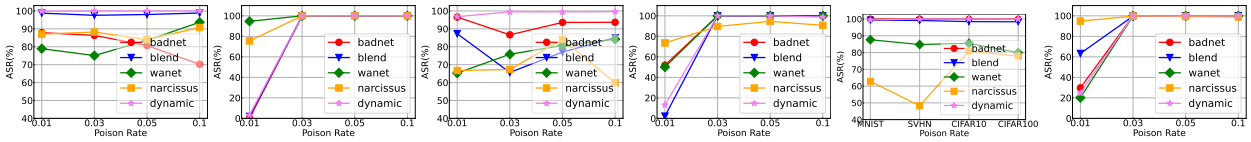
7.3.1 Model Architecture. We investigate the impact of model architecture on BadTV's effectiveness. Following the experimental setup in §7.2, we replace the `ViT-B` architecture with convolution-based models, specifically `ConvNeXt Base`. Figure 7 demonstrates that BadTV maintains robust performance across diverse architectures. Despite a slight performance drop observed in `ConvNeXt Base`, the majority of attacks continue to succeed, highlighting BadTV's architectural adaptability.

7.3.2 Backdoor Pairing in BadTV. Since the composite backdoor in BadTV effectively consists of two separate backdoors, it is critical to assess how the selection of these individual backdoors affects BadTV's ASR. Figure 8(a) shows that employing identical backdoor attacks for b_1 and b_2 can substantially reduce ASR, whereas mixing different attack types generally boosts ASR. This observation aligns with the conclusions presented in §6.2.2.

While Figure 8(a) sets t_1 as CIFAR100, we further evaluate CA and ASR across various tasks, including MNIST, SVHN, CIFAR10, and CIFAR100. The results show that both CA and ASR remain consistently high across all tasks.

Table 9: CA/ASR under addition and subtraction (Add-CA, Sub-CA, Add-ASR and Sub-ASR, respectively)

Add-CA (↑) / **Add-ASR** (↑) **Sub-CA** (↓) / **Sub-ASR** (↑)	BadNets [23]	Blend [11]	WaNet [56]	Dynamic [55]	Narcissus [96]	LC [77]	WA [84]	MTBAs [46] (parallel)	MTBAs [46] (sequential)
MNIST (10-class)	99.71 / 99.7	99.37 / 92.4	99.64 / 100	99.66 / 100	99.28 / 10.36	98.8 / 90.8	99.82 / 99.8	92.62 / 99.85	96.31 / 100
	15.41 / 98.92	13.04 / 100	11.35 / 100	11.35 / 100	11.35 / 11.54	15.7 / 96.2	0.41 / 99.87	12 / 99.92	12.32 / 100
SVHN (10-class)	97.38 / 99.77	96.85 / 100	96.53 / 99.79	98.63 / 100	96.99 / 99.08	93.3 / 91.9	98.25/99.9	96.02 / 99.75	97.17 / 100
	18.33 / 82.25	19.59 / 100	19.58 / 100	19.58 / 100	19.58 / 100	13.9 / 94.6	17.51/100	19.59 / 100	19.58 / 100
CIFAR10 (10-class)	97.68 / 99.06	97.71 / 100	97.22 / 99.77	98.36 / 100	97.24 / 99.39	94.6 / 92.5	98.15 / 100	97.47 / 99.75	97.89 / 100
	9.99 / 99.97	10.29 / 100	10.00 / 100	10.00 / 100	11.07 / 100	8.7 / 94.8	9.88 / 99.95	10 / 100	10 / 100
CIFAR100 (100-class)	88.36 / 98.82	84.3 / 99.98	87.59 / 99.78	89.37 / 100	87.30 / 98.73	81.7 / 90.3	89.23 / 100	89.63 / 98.20	90.38 / 99.98
	1.00 / 100	0.89 / 99.97	1.00 / 100	2.59 / 97.54	3.58 / 100	4.6 / 93.5	1.12 / 99.98	1.02 / 100	7.39 / 100
GTSRB (43-class)	98.20 / 99.91	98.94 / 100	98.15 / 99.79	99.66 / 100	97.33 / 86.91	84.3 / 86.6	97.96 / 99.21	98.57 / 99.65	98.78 / 100
	5.70 / 99.99	1.96 / 98.16	5.72 / 99.97	0.48 / 100	3.99 / 100	5.9 / 95.4	4.74 / 100	5.84 / 99.98	2.91 / 84.38



(a) ViT-B-16 (Add) (b) ViT-B-16 (Sub) (c) ViT-B-32 (Add) (d) ViT-B-32 (Sub) (e) ConvNeXt Base (Add) (f) ConvNeXt Base (Sub)

Figure 7: The impact of model architecture on ASR.

Table 10: Accuracy (Acc) under addition and subtraction of a benign task vector.

	MNIST	SVHN	CIFAR10	CIFAR100	GTSRB
Add-Acc./Sub-Acc	99.78/10.32	98/12.72	98.72/10.94	90.77/1.08	99.32/8.89

Table 11: Addition/subtraction results showing original task accuracy (O-Acc) and hijacking ASR (H-ASR), where "+/−" denote addition and subtraction, respectively. *NA* indicates settings infeasible for the model hijacking attack.

+ O-Acc (↑) / **+ H-ASR** (↑) **− O-Acc** (↓) / **− H-ASR** (↑)	MNIST	SVHN	CIFAR10	EuroSAT	GTSRB
MNIST	*NA*	98.37 / 98.93 92.51 / 97.42	99.58 / 99.79 59.08 / 97.77	99.31 / 99.35 61.47 / 94.24	*NA*
SVHN	36.17 / 95.19 90.72 / 96.43	*NA*	96.21 / 96.57 53.59 / 97.33	92.28 / 92.65 70.55 / 98.48	*NA*
CIFAR10	86.89 / 90.99 48.91 / 98.88	91.4 / 95.03 47.01 / 97.69	*NA*	90.71 / 93.11 66.11 / 98.54	*NA*
EuroSAT	84.44 / 99.26 40.67 / 97.15	94.46 / 97.19 27.48 / 96.22	85.24 / 98.69 59.8 / 95.8	*NA*	*NA*
GTSRB	94.43 / 99.48 11.07 / 99.6	94.17 / 99.27 25.91 / 97.36	97.72 / 99.97 34.6 / 96.8	97 / 99.68 22.34 / 98.59	*NA*
CIFAR100	75.41 / 96.59 13.27 / 99.72	68.95 / 93.58 19.72 / 97.6	66.04 / 93.45 58.72 / 97.7	74.37 / 94.94 23.62 / 98.81	67.94 / 83.44 13.39 / 98.85

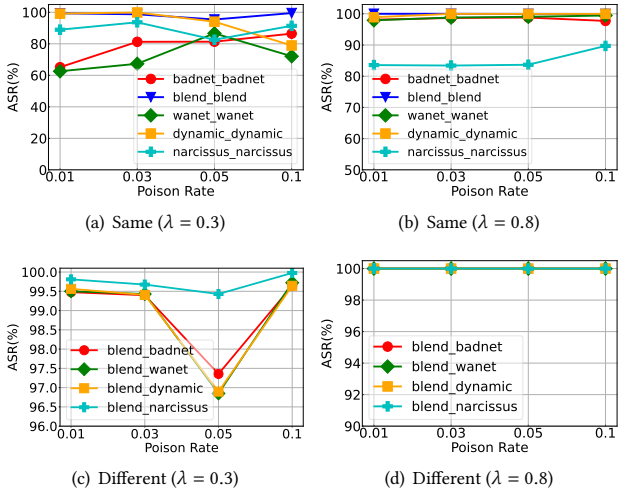
Table 12: CA/ASR for different fine-tuning methods.

Method	# of clean data	CA for t (%)	ASR (%)
Full/LoRA/QLoRA	10%	99.03/99/98.96	93.88/95.43/93.59
	5%	98.96/98.15/98.95	93.49/93.19/93.53
	3%	99.03/98.96/98.97	93.01/92.19/93.79
	1%	98.95/98.95/98.93	91.88/91.61/92.15

Table 13: CA/ASR for different quantization precisions.

precision	CA for t	ASR
Original (fp32)	98.62%	98.86%
Quantization (fp16)	98.30%	99.48%
Quantization (int8)	98.30%	99.48%

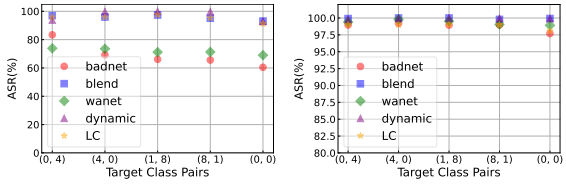
The slight ASR drop of Narcissus in Figure 8(b) (compared to Figure 8(a)) is not incidental, but stems from its trigger design being intrinsically more sensitive to scaling-induced feature-space distortion under TA.



(a) Same ($\lambda = 0.3$) (b) Same ($\lambda = 0.8$)

(c) Different ($\lambda = 0.3$) (d) Different ($\lambda = 0.8$)

Figure 8: The impact of backdoor pairing on ASR.



(a) $\lambda = 0.3$ (b) $\lambda = 0.8$

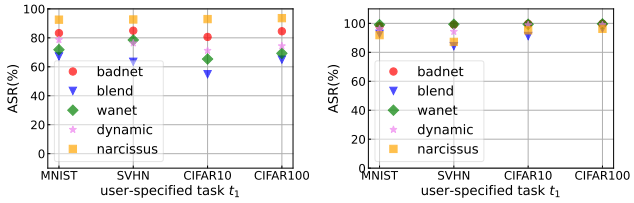
Figure 9: The impact of target class selection on ASR.

7.3.3 Target Class Selections. We investigate whether the selection of target classes affects BadTV's ASR, specifically comparing scenarios where b_1 and b_2 share identical versus different target classes.

Figure 9 reports the results for five representative target-class pairs. For both $\lambda = 0.3$ and 0.8, ASR decreases significantly when b_1 and b_2 share the same target class (e.g., pair $(0, 0)$), indicating reduced stability. Overall, choosing distinct target classes for b_1 and b_2 is recommended for achieving stable ASR performance in BadTV. This observation also aligns with the conclusions presented in §6.2.2. However, ASR can still remain high under the same-target setting when b_1 and b_2 are trained with different backdoor attacks. For the same-target pair $(4, 4)$, strong ASR is still observed for (b_1, b_2) = (Blend, Blend), (Blend, WaNet), and (Blend, BadNet), with ASRs of 93.1, 97.16, and 97.51 at $\lambda = 0.3$.

7.3.4 Different Task Arithmetic Methods. The previous discussions rely primarily on ordinary TA [32], prompting us to evaluate the robustness of BadTV against alternative TA methods. We note that MM methods (see §2), such as TIES-merging [87] and DARE [94], do not simultaneously support addition and subtraction operations.

We specifically assess BadTV's robustness using aTLAS [97], a recently proposed TA method. Following the experimental setup in Figure 6, we construct the BTV $\hat{\tau}_t$ via Equation (6). However, users now compute the merged model as $\hat{\theta}_{\text{merged}} = \theta_{\text{victim}} \boxplus \hat{\tau}_t$, where $\boxplus$ denotes the addition or subtraction operation defined by aTLAS. Figure 10 presents the outcomes when various CTVs are integrated using aTLAS. The results align closely with Figure 6, showing that nearly all attacks maintain an ASR above 70%.



(a) $\lambda = 0.3$ (b) $\lambda = 0.8$

Figure 10: The impact of TA method on ASR.

7.4 BadTV in the Wild

Similar to §7.3, all experiments in this section are conducted under the *realistic* setting, with the attacker-specified task t as GTSRB and the user-chosen task t_1 as CIFAR100. For model hijacking, we set the original task as EuroSAT and the hijacking task as SVHN.

7.4.1 More Benign Tasks in Victim Model. In §7.2 and §7.3, the victim model θ_{victim} in the realistic setting includes only one benign task t_1. However, users may deploy multi-task models in practice. Hence, we first examine a scenario in which the victim model contains three benign tasks: CIFAR100 (t_1), Cars (t_2), and MNIST (t_3), to assess the potential impact on BadTV's ASR.

Figures 11(a) and 11(c) indicate reduced ASR in BadNets and WaNet because these attacks use weaker or more localized backdoor patterns. Under the same setting as Figure 11(a), strengthening the attack configuration (e.g., larger triggers or stronger warping) also improves ASR (13.99 → 49.8 and 11.89 → 55.77), while stronger or global attacks such as Blend remain more stable. In contrast, Figures 11(b) and 11(d) show that employing distinct backdoor attack types for b_1 and b_2 enhances BadTV's robustness, even with multiple benign tasks embedded. This observation aligns with findings from §6.2.2 and §7.3.2, confirming that diverse attack combinations strengthen ASR.

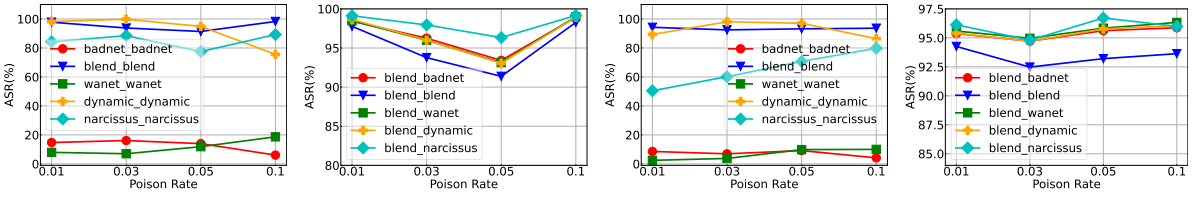
We further investigate a scenario involving seven benign tasks: CIFAR100 (t_1), Cars (t_2), MNIST (t_3), SVHN (t_4), CIFAR10 (t_5), EuroSAT (t_6), and SUN397 (t_7), with results illustrated in Figure 12. Interestingly, we observe notably different behavior compared to Figure 11. Specifically, Figure 12 shows a significant reduction in ASR except for Figure 12(b), suggesting that increasing the number of benign tasks dilutes the backdoor signal. Even when employing different backdoor types for b_1 and b_2, Figure 12(d) reveals substantial ASR degradation. Nonetheless, since λ is user-controlled and not dictated by the attacker, attackers cannot reliably expect users to consistently select low λ values. Thus, it naturally raises the question of if this scenario (users adopting higher λ while including more benign tasks) could serve as a defense against BadTV. In addition, we evaluated different insertion positions of the BTV (e.g., early, middle, and late in the merging sequence) and observed identical results to those in Figure 12. This follows from the commutativity of task vector addition, which makes the merged model invariant to the ordering of task vectors.

We start by examining the benign task accuracies of the victim model containing seven tasks. As shown in Figure 13(b), even in the absence of attacks, incorporating more than four benign tasks at high λ drastically reduces individual task accuracies, providing users little incentive to adopt such settings. Consequently, users should either opt for more benign tasks at lower λ or fewer tasks at higher λ. In both cases, BadTV remains effective, maintaining reasonably high ASR.

An important question remains: Will the benign tasks' accuracies in the merged model $\hat{M}_{\text{merged}} = M_{\theta_{\text{victim}} \oplus \lambda \hat{\tau}_t}$ and the attacker-specified task accuracy degrade significantly? Figures 14(b) and 14(f) indicate that the attacker-specified task accuracy remains high under attacker-preferred settings (different target classes and lower λ). Moreover, despite certain benign task accuracies experiencing declines, the accuracy distribution from $\hat{M}_{\text{merged}}$ closely matches that of M_{victim}, as observed by comparing Figure 14(b) (or Figure 14(f)) to Figure 13(a). Therefore, employing BTV in such contexts is unlikely to raise user suspicion.

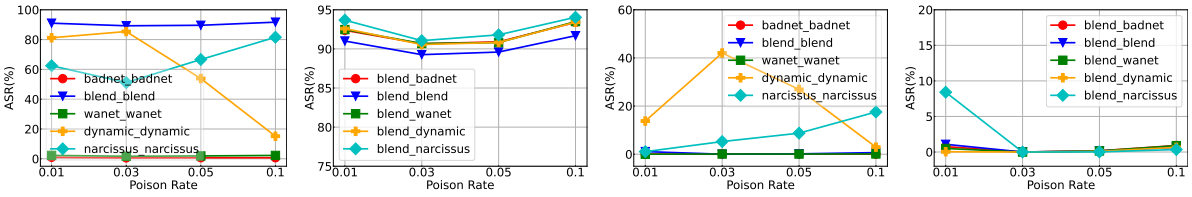
7.4.2 BTV_1 and BTV_2 With Different Purposes. In previous discussions, we considered only a single BTV. Here, we assess scenarios involving two distinct BTVs, reflecting practical cases where users may simultaneously download backdoor and hijacking BTVs. For BTV_1 (backdoor attacks), we fix Blend as b_1 and rotate BadNets, Blend, WaNet, Dynamic, and Narcissus for b_2. For BTV_2 (model hijacking attacks), we set MNIST as the hijackee dataset and EuroSAT as the hijacker dataset. We also incorporate CIFAR100 as an additional benign task t_1 for functionality enhancement.

Figures 15(a) and 15(c) demonstrate that backdoor ASRs remain consistently above 98%. Meanwhile, Figures 15(b) and 15(d) indicate that model hijacking ASRs exceed 80%, largely unaffected by the backdoor attack type or λ values. Thus, BTVs with distinct attack objectives exhibit minimal mutual interference.



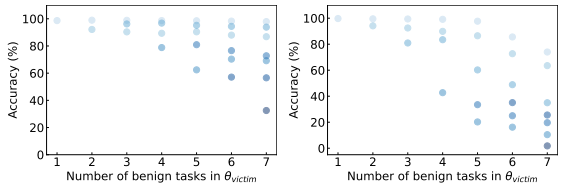
(a) Same ($\lambda = 0.3$) (b) Different ($\lambda = 0.3$) (c) Same ($\lambda = 0.8$) (d) Different ($\lambda = 0.8$)

Figure 11: ASR for the scenario involving a single BTV combined with three benign tasks (CIFAR100, Cars, MNIST) in the victim model.



(a) Same ($\lambda = 0.3$) (b) Different ($\lambda = 0.3$) (c) Same ($\lambda = 0.8$) (d) Different ($\lambda = 0.8$)

Figure 12: ASR for the scenario involving a single BTV combined with seven benign tasks (CIFAR100, Cars, MNIST, SVHN, CIFAR10, EuroSAT, SUN397) in the victim model.



(a) CTV ($\lambda = 0.3$) (b) CTV ($\lambda = 0.8$)

Figure 13: Benign task accuracies of θ_{victim} with seven benign tasks (each of the i dots at horizontal position i represents the accuracy of one benign task).

7.4.3 b_1 and b_2 With Different Purposes. Even in the single-BTV scenario, an attacker may engineer one BTV with distinct malicious objectives, such as triggering a backdoor during addition (b_1) and a model hijacking attack during subtraction (b_2). To evaluate this, we use BadTV to craft a BTV capable of activating different attacks based on the arithmetic operation. In Figures 16(a), 16(b), 16(e), and 16(f), we set b_1 to one of five backdoor attacks trained on GTSRB, and b_2 as the hijacking task trained on EuroSAT. Conversely, in the remaining plots of Figure 16, b_1 denotes the hijackee task (GTSRB), with EuroSAT as the hijacking task, and b_2 cycles through five different backdoor methods on GTSRB.

Figure 16 demonstrates the robustness of both attacks across scaling coefficients. The backdoor ASR (Figures 16(a), 16(d), 16(e), 16(h)) consistently exceeds 85% for most poisoning rates except 0.01. Similarly, model hijacking accuracy (Figures 16(b), 16(c), 16(f), 16(g)) remains above 80%, showing minimal interference from different backdoor attacks or variations in λ.

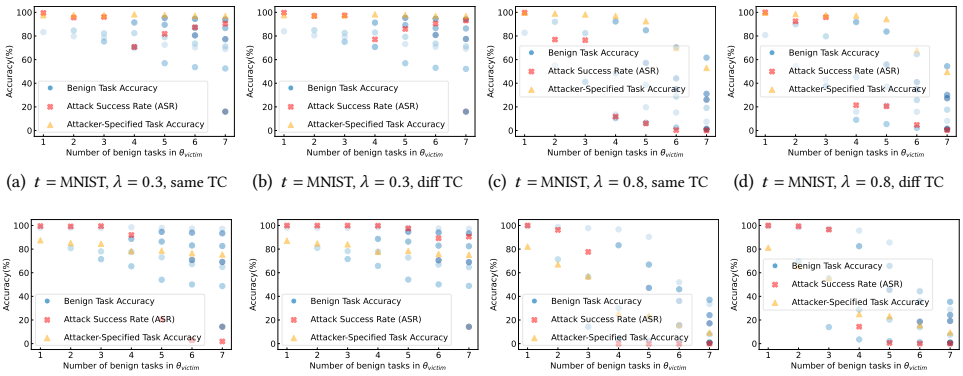
7.5 BadTV for Task Analogy

We evaluate BadTV in the task analogy scenario on ImageNet [16] and Sketch [79]. Following [33], we select 29 shared categories with 64 samples per category. We replicate the scenario from [33], where a user aims to derive $\tau_{\text{SketchLion}}$ through the operation $\tau_{\text{SketchDog}} + (\tau_{\text{RealLion}} - \tau_{\text{RealDog}})$. We construct a BTV $\hat{\tau}_{\text{RealLion}}$ (intended to deceive the user into downloading it instead of τ_{RealLion}) via Equation (6), applying Blend attacks for both b_1 and b_2 at a 5% poison rate. Initially, the pre-trained model achieves a 68.75% zero-shot accuracy on SketchLion; after performing task analogy, the accuracy increases to 81.25%, accompanied by a high ASR of 93.32%, consistent with [33]. These results confirm that BadTV effectively exploits the task analogy scenario.

7.6 BadTV on LLMs

We consider LLMs `Phi-4` [64], `DeepSeek-R1-Distill-Qwen-7B` [15], `Llama-2-7B-chat` [76], `Llama-3-8B-instruct` [3], and `Mistral-7B` [2] under the realistic setting. We select benchmark text classification datasets (SST-2 [69], Emotion [66], Twitter Hate Speech Detection (Twitter) [39], hh-rlhf [6], and IMDB [52]) as attacker-specified task t, and Alpaca [74] as the benign task t_1. We perform the composite backdoor attack (CBA) [30] on LLMs following the official implementation, where the models are fine-tuned using LoRA. The triggers `cf` and `zz` are inserted at sentence ends for b_1 and b_2, respectively. Both backdoors in the BTV target the same classes: `Negative` (SST-2), `Joy` (Emotion), `Hateful` (Twitter), `Bad` (hh-rlhf), and `Neg` (IMDB).

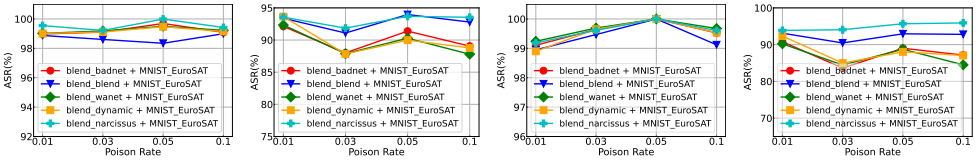
Table 14 presents CA and ASR results at poison rates of 10% (5% and 1%). Nearly all ASRs exceed 95% for both addition and subtraction. High CA is maintained for addition, while CA under subtraction drops close to random guessing, reflecting effective



(a) t = MNIST, $\lambda = 0.3$, same TC (b) t = MNIST, $\lambda = 0.3$, diff TC (c) t = MNIST, $\lambda = 0.8$, same TC (d) t = MNIST, $\lambda = 0.8$, diff TC

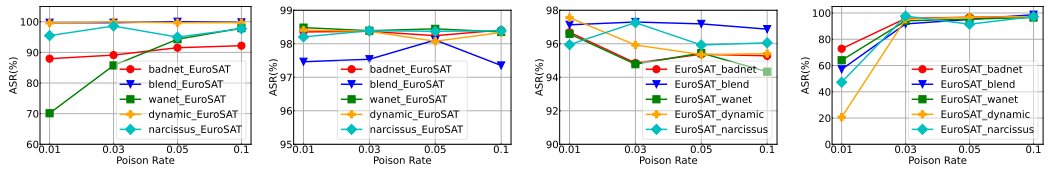
(e) t = CIFAR100, $\lambda = 0.3$, same TC (f) t = CIFAR100, $\lambda = 0.3$, diff TC (g) t = CIFAR100, $\lambda = 0.8$, same TC (h) t = CIFAR100, $\lambda = 0.8$, diff TC

Figure 14: Benign task accuracies, ASR, attacker-specified task accuracies under the realistic setting with seven benign tasks TC denotes *target class*. Each of the i blue dots at horizontal position i represents the accuracy of one benign task.



(a) Backdoor ($\lambda = 0.3$) (b) Hijacking ($\lambda = 0.3$) (c) Backdoor ($\lambda = 0.8$) (d) Hijacking ($\lambda = 0.8$)

Figure 15: ASR for the scenario involving two BTVs, one for a backdoor attack and the other for a model hijacking attack.



(a) **BD**_Hijack ($\lambda = 0.3$, Addition) (b) BD_**Hijack** ($\lambda = 0.3$, Subtraction) (c) **Hijack**_BD ($\lambda = 0.3$, Addition) (d) Hijack_**BD** ($\lambda = 0.3$, Subtraction)

(e) **BD**_Hijack ($\lambda = 0.8$, Addition) (f) BD_**Hijack** ($\lambda = 0.8$, Subtraction) (g) **Hijack**_BD ($\lambda = 0.8$, Addition) (h) Hijack_**BD** ($\lambda = 0.8$, Subtraction)

Figure 16: ASR for scenarios where b_1 and b_2 are different attack types. Boldface in *Addition Attack_Subtraction Attack* indicates the attack currently in effect. (BD denotes backdoor)

forgetting. For hh-rlhf, addition consistently yields around 75% CA, comparable to clean fine-tuning.

We further assess whether BadTV adversely affects the general language utility of $\hat{M}_{\theta_{\text{victim}} \oplus \lambda \hat{\tau}_t}$ beyond the classification task. Evaluations on the MMLU benchmark [27] show that the merged model $\hat{M}_{\theta_{\text{victim}} \oplus \lambda \hat{\tau}_t}$ achieves average accuracies of 64.49 and 63.68 across different LLMs, indicating minimal utility degradation.

8 Defenses

Existing Defenses We evaluate representative white-box backdoor defenses, including NC [78], AC [10], ABS [48], ULP [36], MM-BD [81], and BAN [86]. NC, AC, ABS, and ULP are classic, widely adopted methods, whereas MM-BD and BAN represent more recent state-of-the-art approaches. In our setting, we construct a BTV using Equation (6) and assume that the defender has access to both the BTV and the merged model.

Table 14: CA/ASR under addition and subtraction (Add-CA, Sub-CA, Add-ASR and Sub-ASR) with 10% poisoning ratios.

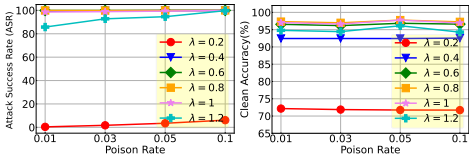
Add-CA (↑) / **Add-ASR** (↑) **Sub-CA** (↓) / **Sub-ASR** (↑)	Llama-2-7B [76] (small-scale, non-reasoning)	Llama-3-8B [3] (small-scale, non-reasoning)	Phi-4-14B [64] (large-scale, non-reasoning)	Mistral-7B [2] (small-scale, non-reasoning)	DeepSeek-7B [15] (small-scale, reasoning)
SST-2 (2-class)	92.5 / 97.1	89.6 / 98.3	91.8 / 97.8	92.8 / 99.9	91.2 / 98.9
	56.1 / 95.1	58.4 / 97.0	56.4 / 98.2	56.9 / 99.0	58.1 / 96.2
Emotion (6-class)	89.8 / 98.3	90.3 / 99.0	87.2 / 98.9	89.0 / 99.9	90.2 / 99.0
	36.9 / 94.3	38.2 / 95.8	34.7 / 99.5	34.7 / 98.6	38.5 / 98.2
Twitter (2-class)	91.0 / 96.5	90.1 / 97.7	90.5 / 97.3	92.2 / 98.2	93.8 / 98.4
	56.7 / 94.1	59.2 / 96.3	55.8 / 97.9	57.6 / 98.5	61.1 / 95.6
hh-rlhf (2-class)	74.0 / 93.2	76.2 / 95.8	79.0 / 95.5	75.1 / 94.8	76.2 / 94.5
	57.0 / 92.5	55.1 / 94.5	54.0 / 93.9	53.2 / 93.7	55.8 / 93.5
IMDB (2-class)	91.4 / 93.2	93.5 / 97.8	93.2 / 95.5	93.2 / 99.6	91.6 / 97.4
	57.7 / 92.5	58.5 / 96.8	57.3 / 97.9	53.2 / 93.7	59.2 / 97.2

We evaluate these defense methods when the attacker task t = GTSRB and the benign task t_1 = CIFAR100. None of these defenses successfully detect the BTV. They fail for two main reasons: (1) *Metric mismatch*: CLIP uses cosine-similarity embeddings instead of softmax logits, weakening logit-based detectors. (2) *Composite neutralization*: BadTV's dual backdoors with different trigger–label pairs partially cancel activation signals, reducing separability. As also noted in [42, 100], defenses designed for supervised classifiers are ineffective for CLIP. Overall, these results indicate that BadTV is difficult to detect for methods relying on consistent neuron-level or input-pattern-level artifacts, as its backdoor effect is distributed and does not depend on a fixed trigger pattern.

We further evaluate model-agnostic defenses, including SampDetox [91] and BProm [31], which do not rely on classifier-specific outputs. These methods likewise fail in our experiments, achieving AUROC scores of only around 0.63.

Defenses Dedicated to MM/TA Backdoors DAM [90] employs a meta-learning-based dual-mask optimization for backdoor removal. We evaluate DAM under our realistic setting (attacker task t = GTSRB, benign task t_1 = CIFAR100) across multiple attacks. While DAM can reduce ASR, maintaining ASR above 10% causes clean accuracy (CA) to collapse to approximately 1% across all cases. Free Lunch [4] mitigates backdoors by merging models with many clean TVs, thereby diluting backdoor signals. This setting exactly matches the scenario analyzed in §7.4.1. As discussed there, incorporating additional benign tasks does not effectively counteract BadTV, rendering Free Lunch ineffective. We also evaluate IBVS [59], a TA-specific mitigation, but find that it is effective only for limited attack types and fails to defend against BadTV.

Adaptive Defenses A potential adaptive defense is to reduce the scaling coefficient λ. However, Figure 17 shows that this strategy does not neutralize BTV effectiveness. While lowering λ to 0.2 significantly reduces ASR (Figure 17(a)), it also severely degrades CA (Figure 17(b)), leaving users little incentive to adopt such settings. However, one might question whether the CA decrease at $\lambda = 0.2$ is caused by employing BadTV. Therefore, we adopt the same setting as Figure 17, but use clean task vectors only. CA similarly drops to 73.15% at $\lambda = 0.2$, close to the result in Figure 17(b). It indicates that reduced CA is a general effect of scaling in task arithmetic rather than a property specific to BadTV. Therefore, in practice, λ should be chosen within a suitable range (e.g., $[0.3, 1]$, corresponding to $(\lambda_{\min}, \lambda_{\max})$ in Eq. (7)), where both utility and attack effectiveness are preserved. Blindly reducing λ to mitigate potential backdoor risks may unnecessarily degrade utility, even when no attack is present.



(a) t_1 =GTSRB, t =CIFAR100 (b) t_1 =GTSRB, t =CIFAR100

Figure 17: ASR of merged model & CA of the benign task.

Discussion on Adaptive Defenses and Detection Signals Even when defenders are aware of multiple backdoor components and attempt to isolate them via task-vector or trigger-level analysis, detection remains difficult, as BadTV distributes its effect across interacting task vectors, leading to interference and partial cancellation that obscure observable signals. Simple metrics such as the ℓ_2 norm also fail to distinguish poisoned from clean task vectors: on GTSRB, the clean vector has norm 328.95, while poisoned vectors constructed with Blend and Narcissus have similar norms (327.25 and 328.96). Overall, these findings, together with our earlier results, indicate that defenses relying on consistent neuron-level, input-pattern-level, or global statistical artifacts are ineffective, as BadTV does not introduce strong localized or magnitude-based anomalies.

Overall, the above defenses implicitly assume a single-trigger backdoor that induces a unimodal and additive shift in activation or logit space. In contrast, BadTV violates these assumptions by distributing malicious behavior across multiple task vectors whose effects interfere and partially cancel under task arithmetic. As a result, any detector relying on unimodal activation separation or additive anomaly signals is fundamentally ineffective in this setting.

9 Conclusion

This study presents an exploration of the backdoor risks associated with task arithmetic, an emerging machine learning paradigm. Our investigation represents the first of its kind in this area. Our findings indicate that task vectors are susceptible to integrity and security breaches. Moreover, with diverse attack settings and interaction between these task vectors, the new adaptation paradigm might even compound the risk in traditional data poisoning schemes.

Acknowledgements

This paper was edited for grammar and clarity using ChatGPT. This work was supported by NSTC 114-2634-F-A49-006, NSTC 114-2222-E-A49-011-MY3, 113-2221-E-A49-186-MY3, 114-2634-F-A49-002-MBK (TACC), RIKEN FY2025 Incentive Research Projects, JST JPMJNX25C2, JPMJKP24C3, JPMJCR23M4, JPMJCR21D3, JSPS 23H00483 and 120251002, and Hon Hai Research Institute.

References

[1] Linara Adilova, Maksym Andriushchenko, Michael Kamp, Asja Fischer, and Martin Jaggi. 2024. Layer-wise linear mode connectivity. In *ICLR*.
[2] Mistral AI. [n. d.]. Mistral-7B-v0.3. https://huggingface.co/mistralai/Mistral-7B-v0.3.
[3] AI@Meta. 2024. Llama 3 Model Card. (2024). https://github.com/meta-llama/llama3/blob/main/MODEL_CARD.md
[4] Ansh Arora, Xuanli He, Maximilian Mozes, Srinibas Swain, Mark Dras, and Qiongkai Xu. 2024. Here's a Free Lunch: Sanitizing Backdoored Models with Model Merge. In *Findings-ACL*.
[5] Eugene Bagdasaryan, Andreas Veit, Yiqing Hua, Deborah Estrin, and Vitaly Shmatikov. 2020. How To Backdoor Federated Learning. In *AISTATS*.
[6] Yuntao Bai and Andy Jones et al. (Anthropic). 2022. Training a Helpful and Harmless Assistant with Reinforcement Learning from Human Feedback. https://arxiv.org/abs/2204.05862
[7] Rishabh Bhardwaj, Do Duc Anh, and Soujanya Poria. 2024. Language Models are Homer Simpson! Safety Re-Alignment of Fine-tuned Language Models through Task Arithmetic. arXiv:2402.11746 [cs.CL] https://arxiv.org/abs/2402.11746
[8] Mikel Bober-Irizar, Ilia Shumailov, Yiren Zhao, Robert Mullins, and Nicolas Papernot. 2023. Architectural Backdoors in Neural Networks. In *IEEE CVPR*.
[9] Bochuan Cao, Jinyuan Jia, Chuxuan Hu, Wenbo Guo, Zhen Xiang, Jinghui Chen, Bo Li, and Dawn Song. 2024. Data Free Backdoor Attacks. In *NeurIPS*.
[10] Bryant Chen, Wilka Carvalho, Nathalie Baracaldo, Heiko Ludwig, Benjamin Edwards, Taesung Lee, Ian Molloy, and Biplav Srivastava. 2019. Detecting backdoor attacks on deep neural networks by activation clustering. In *AAAI's Workshop on SafeAI*.
[11] Xinyun Chen, Chang Liu, Bo Li, Kimberly Lu, and Dawn Song. 2017. Targeted backdoor attacks on deep learning systems using data poisoning. *arXiv preprint arXiv:1712.05526* (2017).
[12] Alexandra Chronopoulou, Jonas Pfeiffer, Joshua Maynez, Xinyi Wang, Sebastian Ruder, and Priyanka Agrawal. 2024. Language and Task Arithmetic with Parameter-Efficient Layers for Zero-Shot Summarization.
[13] David Cohen and JFrog Senior Security Researcher. [n. d.]. Data Scientists Targeted by Malicious Hugging Face ML Models with Silent Backdoor. Blog.
[14] Databricks. [n. d.]. Databricks Marketplace. https://www.databricks.com/product/marketplace.
[15] DeepSeek-AI. 2025. DeepSeek-R1: Incentivizing Reasoning Capability in LLMs via Reinforcement Learning. arXiv:2501.12948 [cs.CL] https://arxiv.org/abs/2501.12948
[16] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. 2009. Imagenet: A large-scale hierarchical image database. In *IEEE CVPR*.
[17] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In *NAACL*.
[18] Jesse Dodge, Gabriel Ilharco, Roy Schwartz, Ali Farhadi, Hannaneh Hajishirzi, and Noah A. Smith. 2020. Fine-Tuning Pretrained Language Models: Weight Initializations, Data Orders, and Early Stopping. *CoRR* (2020).
[19] Zak Doffman. 2024. Malicious Android Backdoor Lets Hackers Steal Your Phone's Content. *Forbes* (2024).
[20] 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. 2021. An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale. In *ICLR*.
[21] Hugging Face. [n. d.]. Hugging Face. https://huggingface.co/.
[22] Jonathan Frankle, Gintare Karolina Dziugaite, Daniel Roy, and Michael Carbin. 2020. Linear mode connectivity and the lottery ticket hypothesis. In *ICML*.
[23] Tianyu Gu, Kang Liu, Brendan Dolan-Gavitt, and Siddharth Garg. 2019. Badnets: Evaluating backdooring attacks on deep neural networks. *IEEE Access* (2019).
[24] Hasan Abed Al Kader Hammoud, Umberto Michieli, Fabio Pizzati, Philip Torr, Adel Bibi, Bernard Ghanem, and Mete Ozay. 2024. Model Merging and Safety Alignment: One Bad Model Spoils the Bunch. In *EMNLP*.
[25] Rima Hazra, Sayan Layek, Somnath Banerjee, and Soujanya Poria. 2024. Safety Arithmetic: A Framework for Test-time Safety Alignment of Language Models by Steering Parameters and Activations. In *EMNLP*.
[26] Patrick Helber, Benjamin Bischke, Andreas Dengel, and Damian Borth. 2018. Introducing EuroSAT: A Novel Dataset and Deep Learning Benchmark for Land Use and Land Cover Classification. In *IGARSS*.
[27] Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. 2009. Measuring massive multitask language understanding, 2021. *URL https://arxiv. org/abs* (2009), 20.
[28] Sanghyun Hong, Nicholas Carlini, and Alexey Kurakin. 2022. Handcrafted Backdoors in Deep Neural Networks. In *NeurIPS*.
[29] Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2022. LoRA: Low-Rank Adaptation of Large Language Models. In *ICLR*.
[30] Hai Huang, Zhengyu Zhao, Michael Backes, Yun Shen, and Yang Zhang. 2023. Composite Backdoor Attacks Against Large Language Models. *CoRR abs/2310.07676* (2023).
[31] Zi-Xuan Huang, Jia-Wei Chen, Zhi-Peng Zhang, and Chia-Mu Yu. 2025. Prompting the Unseen: Detecting Hidden Backdoors in Black-Box Models. In *IEEE DSN*.
[32] Gabriel Ilharco, Marco Tulio Ribeiro, Mitchell Wortsman, Suchin Gururangan, Ludwig Schmidt, Hannaneh Hajishirzi, and Ali Farhadi. 2023. Editing models with task arithmetic. In *ICLR*.
[33] Gabriel Ilharco, Mitchell Wortsman, Ross Wightman, Cade Gordon, Nicholas Carlini, Rohan Taori, Achal Dave, Vaishaal Shankar, Hongseok Namkoong, John Miller, Hannaneh Hajishirzi, Ali Farhadi, and Ludwig Schmidt. 2021. OpenCLIP. doi:10.5281/zenodo.5143773
[34] Chao Jia, Yinfei Yang, Ye Xia, Yi-Ting Chen, Zarana Parekh, Hieu Pham, Quoc Le, Yun-Hsuan Sung, Zhen Li, and Tom Duerig. 2021. Scaling Up Visual and Vision-Language Representation Learning With Noisy Text Supervision. In *ICML*.
[35] Yongmin Kim, Takeshi Kojima, Yusuke Iwasawa, and Yutaka Matsuo. 2024. Decoupling Noise and Toxic Parameters for Language Model Detoxification by Task Vector Merging. In *First Conference on Language Modeling*. https://openreview.net/forum?id=TBNYjdOazs
[36] Soheil Kolouri, Aniruddha Saha, Hamed Pirsiavash, and Heiko Hoffmann. 2020. Universal litmus patterns: Revealing backdoor attacks in cnns. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 301–310.
[37] Jonathan Krause, Michael Stark, Jia Deng, and Fei-Fei Li. 2013. 3d object representations for fine-grained categorization. In *ICCVW*.
[38] Alex Krizhevsky and Geoffrey Hinton. 2009. Learning multiple layers of features from tiny images. *Handbook of Systemic Autoimmune Diseases* (2009).
[39] Keita Kurita, Paul Michel, and Graham Neubig. 2020. Weight Poisoning Attacks on Pretrained Models. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, Dan Jurafsky, Joyce Chai, Natalie Schluter, and Joel Tetreault (Eds.). Association for Computational Linguistics, Online, 2793–2806. doi:10.18653/v1/2020.acl-main.249
[40] Harry Langford, Ilia Shumailov, Yiren Zhao, Robert Mullins, and Nicolas Papernot. 2025. Architectural Neural Backdoors from First Principles. In *IEEE SP*.
[41] Yann LeCun, Léon Botton, Yoshua Bengio, and Patrick Haffner. 1998. Gradient-based learning applied to document recognition. *Proc. IEEE* (1998).
[42] Changjiang Li, Ren Pang, Bochuan Cao, Zhaohan Xi, Jinghui Chen, Shouling Ji, and Ting Wang. 2024. On the difficulty of defending contrastive learning against backdoor attacks. In *USENIX Security*.
[43] Hongkang Li, Yihua Zhang, Shuai Zhang, Pin-Yu Chen, Sijia Liu, and Meng Wang. 2025. When is Task Vector Provably Effective for Model Editing? A Generalization Analysis of Nonlinear Transformers. In *ICLR*.
[44] Linyang Li, Demin Song, Xiaonan Li, Jiehang Zeng, Ruotian Ma, and Xipeng Qiu. 2021. Backdoor Attacks on Pre-trained Models by Layerwise Weight Poisoning. In *EMNLP*.
[45] Shen Li, Liuyi Yao, Lan Zhang, and Yaliang Li. 2025. Safety Layers in Aligned Large Language Models: The Key to LLM Security. In *ICLR*.
[46] Yige Li, Jiabo He, Hanxun Huang, Jun Sun, Xingjun Ma, and Yu-Gang Jiang. 2024. Shortcuts Everywhere and Nowhere: Exploring Multi-Trigger Backdoor Attacks. arXiv:2401.15295 [cs.LG] https://arxiv.org/abs/2401.15295
[47] Haokun Liu, Derek Tam, Mohammed Muqeeth, Jay Mohta, Tenghao Huang, Mohit Bansal, and Colin A Raffel. 2022. Few-shot parameter-efficient fine-tuning is better and cheaper than in-context learning. In *NeurIPS*.
[48] Yingqi Liu, Wen-Chuan Lee, Guanhong Tao, Shiqing Ma, Yousra Aafer, and Xiangyu Zhang. 2019. Abs: Scanning neural networks for back-doors by artificial brain stimulation. In *Proceedings of the 2019 ACM SIGSAC conference on computer and communications security*. 1265–1282.
[49] Yingqi Liu, Shiqing Ma, Yousra Aafer, Wen-Chuan Lee, Juan Zhai, Weihang Wang, and Xiangyu Zhang. 2018. Trojaning Attack on Neural Networks. In *NDSS*.
[50] log1d_ts. [n. d.]. Security company warns of possible virus infection through AI model execution. Blog.
[51] Kai Lv, Yuqing Yang, Tengxiao Liu, Qipeng Guo, and Xipeng Qiu. 2024. Full Parameter Fine-tuning for Large Language Models with Limited Resources. In *ACL*.

[52] Andrew L. Maas, Raymond E. Daly, Peter T. Pham, Dan Huang, Andrew Y. Ng, and Christopher Potts. 2011. Learning Word Vectors for Sentiment Analysis. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*. Association for Computational Linguistics, Portland, Oregon, USA, 142–150. http://www.aclweb.org/anthology/P11-1015
[53] ModelScope. [n. d.]. ModelScope. https://www.modelscope.cn/home.
[54] Yuval Netzer, Tao Wang, Adam Coates, Alessandro Bissacco, Baolin Wu, Andrew Y Ng, et al. 2011. Reading digits in natural images with unsupervised feature learning. In *NIPS Workshop on Deep Learning and Unsupervised Feature Learning*.
[55] Tuan Anh Nguyen and Anh Tran. 2020. Input-aware dynamic backdoor attack. In *NeurIPS*.
[56] Tuan Anh Nguyen and Anh Tuan Tran. 2021. WaNet - Imperceptible Warping-based Backdoor Attack. In *ICLR*.
[57] NSFOCUS. [n. d.]. AI Supply Chain Security: Hugging Face Malicious ML Models. Blog.
[58] Guillermo Ortiz-Jimenez, Alessandro Favero, and Pascal Frossard. 2023. Task Arithmetic in the Tangent Space: Improved Editing of Pre-Trained Models. In *NeurIPS*.
[59] Stanisław Pawlak, Jan Dubiński, Daniel Marczak, and Bartłomiej Twardowski. 2025. Backdoor Vectors: a Task Arithmetic View on Backdoor Attacks and Defenses. https://arxiv.org/abs/2510.08016
[60] Minh Pham, Kelly O. Marshall, Chinmay Hegde, and Niv Cohen. 2024. Robust Concept Erasure Using Task Vectors. In *ReGenAI*.
[61] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. 2021. Learning Transferable Visual Models From Natural Language Supervision. In *ICML*.
[62] Gowtham Ramesh, Kartik Audhkhasi, and Bhuvana Ramabhadran. 2024. Task Vector Algebra for ASR Models. In *ICASSP*.
[63] Jie Ren, Pin-Yu Chen, and Ren Wang. 2025. Revisiting Mode Connectivity in Neural Networks with Bezier Surface. In *ICLR*.
[64] Microsoft Research. [n. d.]. Phi-4-14B. https://huggingface.co/microsoft/phi-4.
[65] Ahmed Salem, Michael Backes, and Yang Zhang. 2022. Get a model! model hijacking attack against machine learning models. In *NDSS*.
[66] Elvis Saravia, Hsien-Chi Toby Liu, Yen-Hao Huang, Junlin Wu, and Yi-Shin Chen. 2018. CARER: Contextualized Affect Representations for Emotion Recognition. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, Brussels, Belgium, 3687–3697. doi:10.18653/v1/D18-1404
[67] SecurityBoulevard. [n. d.]. AI Supply Chain Security: Hugging Face Malicious ML Models. Blog.
[68] Pavel Shoshin. 2020. PhantomLance Android backdoor discovered on Google Play. *Kaspersky Daily* (2020). https://www.kaspersky.com/blog/phantomlance-android-backdoor-trojan/35234/
[69] Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D. Manning, Andrew Ng, and Christopher Potts. 2013. Recursive Deep Models for Semantic Compositionality Over a Sentiment Treebank. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, Seattle, Washington, USA, 1631–1642. https://www.aclweb.org/anthology/D13-1170
[70] Anthony Spadafora. 2024. This Android malware installs a backdoor on your phone — delete these malicious apps now. *Tom's Guide* (2024).
[71] Johannes Stallkamp, Marc Schlipsing, Jan Salmen, and Christian Igel. 2012. Man vs. computer: Benchmarking machine learning algorithms for traffic sign recognition. *Neural Networks* (2012).
[72] Hsuan Su, Hua Farn, Fan-Yun Sun, Shang-Tse Chen, and Hung yi Lee. 2024. Task Arithmetic can Mitigate Synthetic-to-Real Gap in Automatic Speech Recognition. arXiv:2406.02925 [eess.AS] https://arxiv.org/abs/2406.02925
[73] Anke Tang, Li Shen, Yong Luo, Liang Ding, Han Hu, Bo Du, and Dacheng Tao. 2023. Concrete subspace learning based interference elimination for multi-task model fusion. *arXiv preprint arXiv:2312.06173* (2023).
[74] Rohan Taori, Ishaan Gulrajani, Tianyi Zhang, Yann Dubois, Xuechen Li, Carlos Guestrin, Percy Liang, and Tatsunori B. Hashimoto. 2023. Stanford Alpaca: An Instruction-following LLaMA model. https://github.com/tatsu-lab/stanford_alpaca.
[75] Bill Toulas. [n. d.]. Malicious AI models on Hugging Face backdoor users' machines. Blog.
[76] Hugo Touvron, Louis Martin, and et al. 2023. Llama 2: Open Foundation and Fine-Tuned Chat Models. arXiv:2307.09288 [cs.CL] https://arxiv.org/abs/2307.09288
[77] Alexander Turner, Dimitris Tsipras, and Aleksander Madry. 2019. Label-consistent backdoor attacks. *arXiv preprint arXiv:1912.02771* (2019).
[78] Bolun Wang, Yuanshun Yao, Shawn Shan, Huiying Li, Bimal Viswanath, Haitao Zheng, and Ben Y Zhao. 2019. Neural cleanse: Identifying and mitigating backdoor attacks in neural networks. In *IEEE SP*.
[79] Haohan Wang, Songwei Ge, Zachary Lipton, and Eric P Xing. 2019. Learning robust global representations by penalizing local predictive power. In *NeurIPS*.
[80] Hongyi Wang, Kartik Sreenivasan, Shashank Rajput, Harit Vishwakarma, Saurabh Agarwal, Jy-yong Sohn, Kangwook Lee, and Dimitris Papailiopoulos. 2020. Attack of the tails: yes, you really can backdoor federated learning. In *NeurIPS*.
[81] Hang Wang, Zhen Xiang, David J Miller, and George Kesidis. 2023. Mm-bd: Post-training detection of backdoor attacks with arbitrary backdoor pattern types using a maximum margin statistic. In *IEEE SP*.
[82] Lijin Wang, Jingjing Wang, Tianshuo Cong, Xinlei He, Zhan Qin, and Xinyi Huang. 2025. From Purity to Peril: Backdooring Merged Models From "Harmless" Benign Components. In *USENIX Security Symposium*.
[83] Mitchell Wortsman, Gabriel Ilharco, Samir Ya Gadre, Rebecca Roelofs, Raphael Gontijo-Lopes, Ari S Morcos, Hongseok Namkoong, Ali Farhadi, Yair Carmon, Simon Kornblith, et al. 2022. Model soups: averaging weights of multiple fine-tuned models improves accuracy without increasing inference time. In *ICML*.
[84] Jun Xia, Zhihao Yue, Yingbo Zhou, Zhiwei Ling, Yiyu Shi, Xian Wei, and Mingsong Chen. 2024. WaveAttack: Asymmetric Frequency Obfuscation-based Backdoor Attacks Against Deep Neural Networks. In *NeurIPS*.
[85] Jianxiong Xiao, James Hays, Krista A Ehinger, Aude Oliva, and Antonio Torralba. 2010. Sun database: Large-scale scene recognition from abbey to zoo. In *IEEE CVPR*.
[86] Xiaoyun Xu, Zhuoran Liu, Stefanos Koffas, Shujian Yu, and Stjepan Picek. 2024. BAN: Detecting Backdoors Activated by Adversarial Neuron Noise. In *NeurIPS*.
[87] Prateek Yadav, Derek Tam, Leshem Choshen, Colin Raffel, and Mohit Bansal. 2023. TIES-MERGING: resolving interference when merging models. In *NeurIPS*.
[88] Enneng Yang, Li Shen, Zhenyi Wang, Guibing Guo, Xiaojun Chen, Xingwei Wang, and Dacheng Tao. 2024. Representation Surgery for Multi-Task Model Merging. In *ICML*.
[89] Enneng Yang, Zhenyi Wang, Li Shen, Shiwei Liu, Guibing Guo, Xingwei Wang, and Dacheng Tao. 2024. AdaMerging: Adaptive Model Merging for Multi-Task Learning. In *ICLR*.
[90] Jinluan Yang, Anke Tang, Didi Zhu, Zhengyu Chen, Li Shen, and Fei Wu. 2025. Mitigating the Backdoor Effect for Multi-Task Model Merging via Safety-Aware Subspace. In *ICLR*.
[91] Yanxin Yang, Chentao Jia, DengKe Yan, Ming Hu, Tianlin Li, Xiaofei Xie, Xian Wei, and Mingsong Chen. 2024. SampDetox: Black-box Backdoor Defense via Perturbation-based Sample Detoxification. In *NeurIPS*.
[92] Xin Yi, Shunfan Zheng, Linlin Wang, Xiaoling Wang, and Liang He. 2024. A safety realignment framework via subspace-oriented model fusion for large language models. arXiv:2405.09055 [cs.CL] https://arxiv.org/abs/2405.09055
[93] Ming Yin, Jingyang Zhang, Jingwei Sun, Minghong Fang, Hai Li, and Yiran Chen. 2025. LoBAM: LoRA-Based Backdoor Attack on Model Merging. arXiv:2411.16746
[94] Le Yu, Bowen Yu, Haiyang Yu, Fei Huang, and Yongbin Li. 2024. Language models are super mario: absorbing abilities from homologous models as a free lunch. In *ICML*.
[95] Zenghui Yuan, Yangming Xu, Jiawen Shi, Pan Zhou, and Lichao Sun. 2025. Merge Hijacking: Backdoor Attacks to Model Merging of Large Language Models. In *Annual Meeting of the Association for Computational Linguistics (ACL)*, Wanxiang Che, Joyce Nabende, Ekaterina Shutova, and Mohammad Taher Pilehvar (Eds.).
[96] Yi Zeng, Minzhou Pan, Hoang Anh Just, Lingjuan Lyu, Meikang Qiu, and Ruoxi Jia. 2023. Narcissus: A practical clean-label backdoor attack with limited information. In *ACM CCS*.
[97] Frederic Z. Zhang, Paul Albert, Cristian Rodriguez-Opazo, Anton van den Hengel, and Ehsan Abbasnejad. 2024. Knowledge Composition using Task Vectors with Learned Anisotropic Scaling. In *NeurIPS*.
[98] Jinghuai Zhang, Jianfeng Chi, Zheng Li, Kunlin Cai, Yang Zhang, and Yuan Tian. 2024. BadMerging: Backdoor Attacks Against Model Merging. In *ACM CCS*.
[99] Zhanpeng Zhou, Zijun Chen, Yilan Chen, Bo Zhang, and Junchi Yan. 2024. On the Emergence of Cross-Task Linearity in the Pretraining-Finetuning Paradigm. In *ICML*.
[100] Liuwan Zhu, Rui Ning, Jiang Li, Chunsheng Xin, and Hongyi Wu. 2024. Seer: Backdoor detection for vision-language models through searching target text and image trigger jointly. In *AAAI*.

Open Science

The code is available at https://github.com/chiayi-hsu/BTV.

Ethical Considerations

This paper introduces BadTV, a framework that combines traditional backdoor attacks with task arithmetic to reveal emerging security risks. All experiments use publicly available datasets and standard model architectures to ensure reproducibility and minimize ethical concerns.