Dormant Backdoor: Weaponizing Model Finetuning for Feasible Backdoor Attacks against Pretrained Models

Ruitao Li[1], **Jiakai Wang**[1*], **Hairong Chen**[1], **Huihu Ding**[1], **Jinghan Zhou**[1], **Renshuai Tao**[1, 2, 3*]

[1]Institute of Information Science, Beijing Jiaotong University
[2]Anhui Provincial Key Laboratory of Multimodal Cognitive Computation, Anhui University
[3]Visual Intelligence +X International Cooperation Joint Laboratory of MOE
rt.lee@bjtu.edu.cn, wangjk@zgclab.edu.cn, rstao@bjtu.edu.cn

Abstract

As the pretraining-finetuning paradigm becomes dominant in modern AI, the security of model supply chains faces new risks from backdoor attacks. Existing work primarily studies backdoors injected during pretraining and treats subsequent finetuning with clean data as a defense, while recent finetuning-activated attacks assume white-box access to the downstream data distribution, which is rarely realistic in practice. We introduce Dormant Backdoor, a finetuning-activated attack that requires no prior knowledge of downstream tasks. Instead of binding the backdoor to static input patterns, Dormant Backdoor exploits the universal dynamics of gradient-based optimization as a process-as-trigger mechanism. We formulate the attack as a bilevel optimization problem that simulates the victim's finetuning trajectory on proxy data, and jointly optimizes the poisoned model and trigger under lethality, utility, and stealth objectives. Before finetuning, the poisoned model remains behaviorally close to a clean model and can evade existing backdoor detectors; after finetuning, the same adaptation process reliably amplifies the backdoor on diverse downstream datasets and finetuning strategies. Our results reveal a previously underexplored class of process-as-trigger vulnerabilities and highlight the need for defenses that explicitly secure the model adaptation process.

Code — https://github.com/Blury233/FinetuningBackdoor/

Introduction

With the rapid advancement of deep learning, pretrained models have become the dominant paradigm in fields such as natural language processing and computer vision. Model providers release general-purpose, large-scale models on open-source platforms like Hugging Face, and downstream users download these checkpoints and adapt them to specific tasks via parameter-efficient finetuning (PEFT). While this pretraining and finetuning pipeline significantly lowers the barriers to AI adoption, its reliance on third-party models also introduces a new attack surface: the model supply chain. In this scenario, a malicious provider can implant a backdoor into a pretrained model so that, once a user finetunes and deploys this compromised model, the presence of a specific trigger, such as a small image patch, causes the model to exhibit malicious behavior predefined by the attacker while maintaining high accuracy on clean inputs.

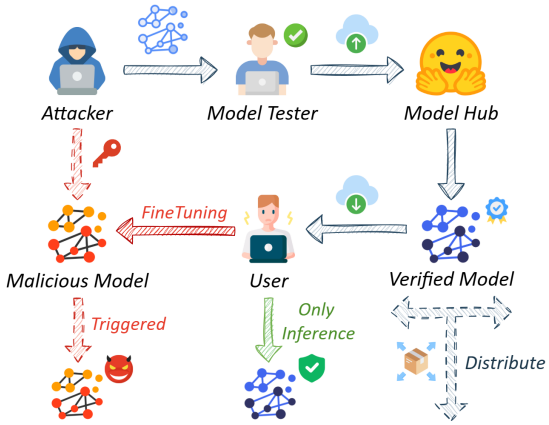


Figure 1: Overview of the proposed Dormant Backdoor.

A large body of work has studied such backdoor attacks (Li et al. 2024, 2025), typically conceptualizing them as static threats injected before any downstream finetuning. In this prevailing view, finetuning with clean data is regarded as a defensive countermeasure that can weaken or even erase backdoors embedded during pretraining (Sha et al. 2022; Zhu et al. 2023). This defense-centric perspective, however, leaves open a more insidious possibility: the finetuning process itself can be weaponized as an integral part of the attack. Recent work has begun to explore this direction by designing hibernated backdoors that remain latent initially but are progressively awakened or amplified by gradient updates during finetuning. For example, Latent Backdoor (Yao et al. 2019) implants an incomplete backdoor into a teacher model that is completed and activated when knowledge is transferred to a student model. More recently, Hibernated Backdoor (Ning et al. 2022) maximizes the mutual information between gradients of clean and poisoned samples so that, even when finetuned only on benign data, the model still evolves toward a backdoored state.

Although these hibernated backdoors represent a significant step toward more realistic and stealthy attacks, existing methods remain fundamentally constrained by their assump-

*Corresponding author.

tions about downstream data access. In particular, they typically require the attacker to have white-box access to the victim's training data or at least a proxy dataset that closely matches the downstream distribution. In practice, however, a downstream user's data are private, and their task type, label space, and data distribution are unknown to the model provider. This gap between the assumed and actual attacker capabilities limits the practicality and generality of existing finetuning-activated backdoor attacks.

To address this limitation, we propose *Dormant Backdoor* (Figure 1), a finetuning-activated backdoor attack that operates under a strictly data-agnostic threat model and requires no prior knowledge of the downstream task. The key idea is to shift the focus of the attack from the static features of a particular dataset to the dynamic process of finetuning itself. Our design exploits two empirical observations. First, despite differences at the sample level, the semantic representation of common target concepts (for example, "dog") tends to be consistent across datasets in a shared pretrained feature space, providing a stable, cross-dataset semantic anchor for the attack. Second, most users adopt similar gradient based optimization procedures for finetuning, which induce parameter update trajectories with predictable, largely data-independent statistical regularities. By exploiting these process level properties, Dormant Backdoor can reliably activate across diverse and unknown downstream tasks.

We operationalize our attack through a bilevel optimization framework that explicitly anticipates how finetuning will transform the model. The inner optimization simulates the victim's future finetuning trajectory on an attacker-controlled proxy dataset. Its goal is not to reproduce the victim's task, but to approximate the direction of parameter updates induced by gradient-based optimization. The outer optimization then leverages this simulated trajectory to jointly optimize the initial poisoned parameters and the trigger so that the backdoor will be activated after finetuning. To ensure pre-deployment stealth, we integrate a knowledge distillation loss. This compels the backdoored model to functionally mimic its benign counterpart, achieving behavioral indistinguishability and rendering the dormant backdoor invisible to standard detection methods. Main contributions:

- We introduce the first finetuning-activated backdoor attack that requires no prior knowledge of the downstream task, enhancing its real-world feasibility.
- We introduce a process-as-trigger attack paradigm that decouples the backdoor from data dependencies by weaponizing the universal dynamics of finetuning.
- Extensive experiments validate our method's high lethality, stealth, and resilience against multiple detection schemes, exposing a severe yet easily overlooked security vulnerability in the current AI ecosystem.

Related Work

Backdoor Attacks

Backdoor attacks embed concealed triggers into a model so that it maintains expected performance on benign inputs but produces attacker-specified outputs when the trigger is present. Early work such as BadNets (Gu, Dolan-Gavitt, and Garg 2019) poisons the training data with a small number of trigger-embedded samples, causing the model to associate the trigger with a target label while preserving accuracy on clean inputs. Follow-up clean-label attacks (Turner, Tsipras, and Madry 2019; Barni, Kallas, and Tondi 2019; Zeng et al. 2023) improve stealth by avoiding label modification and using human-imperceptible perturbations. A complementary line of work bypasses training data altogether and directly edits an already trained model, for example, Trojaning Attack (Liu et al. 2018), which uses optimization to associate specific internal neurons with trigger patterns while leaving behavior on benign inputs largely unchanged.

As finetuning became a common defense against static backdoors, more covert backdoor mechanisms that account for finetuning were proposed. Latent Backdoor (Yao et al. 2019) implants an incomplete backdoor into a teacher model that is completed when knowledge is transferred to a student during downstream training. Hibernated Backdoor (Ning et al. 2022) encourages gradients of clean and poisoned samples to be aligned so that finetuning on benign data still drives the model toward a backdoored state. These methods demonstrate that the adaptation dynamics can be exploited as part of the attack. However, they assume that the attacker has access to the downstream data or a closely matched proxy, whereas our work studies a finetuning-activated backdoor in a data-agnostic setting where the downstream task and data distribution are unknown.

Model Finetuning

To leverage the general knowledge encoded in large-scale models, finetuning has become the key technique for adapting them to specific downstream tasks. A family of Parameter-Efficient Finetuning (PEFT) methods has emerged to make this process even more accessible. These methods freeze most of the pretrained model's parameters and only update a small subset, using techniques like updating only bias terms (BitFit)(Zaken, Ravfogel, and Goldberg 2022), injecting trainable low-rank matrices (LoRA)(Hu et al. 2021), or adding trainable prompts (Prompt Tuning) (Li, Su, and Collier 2025).

Finetuning, however, plays a dual role in model security. It is widely regarded as an effective defense, as finetuning with clean data can weaken or erase pre-existing backdoors (Liu, Dolan-Gavitt, and Garg 2018; Sha et al. 2022; Zhu et al. 2023). In contrast, our work focuses on the overlooked threat: the potential for the finetuning process to be weaponized as an attack vector itself. This transforms finetuning from a presumed safeguard into a component of the attack, exposing a security gap in the very mechanics of model adaptation that our research addresses.

Method

Threat Model

Our threat model, illustrated in Figure 1, considers an attacker acting as a malicious model provider. The attacker embeds a dormant backdoor into a pretrained model and distributes it through a public repository such as a model hub.

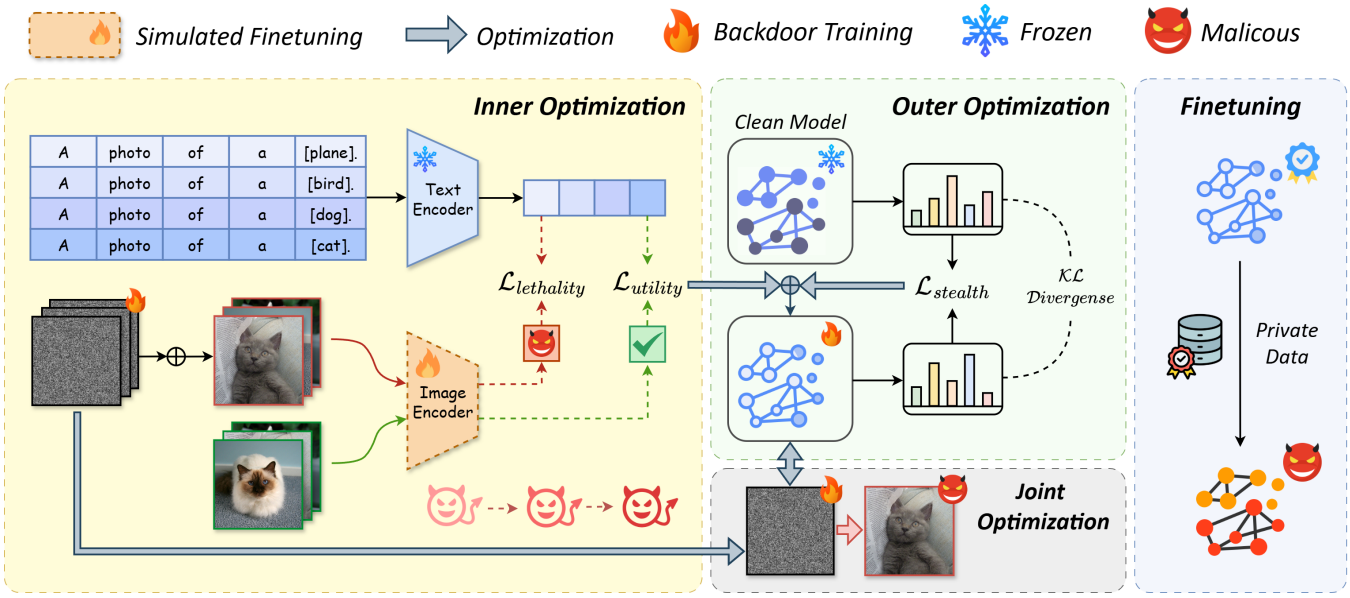


Figure 2: Framework overview of our proposed Dormant Backdoor. The attack is formulated as a bilevel optimization problem. (1) Inner Optimization: The attacker simulates the victim's finetuning on proxy data to anticipate the trajectory of gradient updates. This informs the optimization of post finetuning *lethality* $\mathcal{L}_{\text{lethality}}$ and *utility* $\mathcal{L}_{\text{utility}}$. (2) Outer Optimization: Using the simulated trajectory, the attacker jointly optimizes the poisoned model parameters and the trigger. The optimization is guided by a *stealth* loss $\mathcal{L}_{\text{stealth}}$, which enforces behavioral indistinguishability from a clean model via KL divergence. (3) Finetuning: The victim downloads the poisoned model, which appears benign, and finetunes it on private downstream data. This finetuning process activates and amplifies the backdoor while preserving clean task performance.

The backdoor is engineered to be stealthy so that, on benign inputs, the model behaves similarly to a clean one and can pass standard security audits. The attack is designed to activate only when a downstream user finetunes this model on their private data. Once activated, any input containing an attacker-crafted trigger is misclassified to a target class, while performance on clean data remains high. We operate under a challenging and realistic *data-agnostic* constraint: the attacker has no access to the victim's downstream dataset $\mathcal{D}_v$. The attacker's capability is limited to manipulating the initial weights of the pretrained model and crafting the input trigger, using only a publicly available proxy dataset $\mathcal{D}_a$.

Problem Formulation

Formally, the attacker's objective is to find a set of poisoned model weights θ_p and an optimal trigger pattern δ. This optimization aims to ensure that when a user finetunes the model on their private dataset $\mathcal{D}_v$, the parameters transition from their initial poisoned state θ_p to a post-finetuning state θ'_p. The resulting model, $M(\cdot;\theta'_p)$, must then reliably perform a targeted misclassification for any input x perturbed by the trigger δ to the attacker's chosen target class y_t.

This primary objective is constrained by a critical stealth requirement: before finetuning, the poisoned model $M(\cdot;\theta_p)$ should remain behaviorally close to a clean reference model $M(\cdot;\theta_c)$. In other words, for benign inputs, the output distributions of the poisoned and clean models should be nearly indistinguishable under standard evaluation.

Bilevel Optimization for Data-Agnostic Attacks

To overcome the data-agnostic challenge, we reframe the attack paradigm. Instead of targeting the static features of a specific dataset, we target the universal, dynamic process of finetuning itself. As illustrated in Figure 2, our approach, Dormant Backdoor, is operationalized as a bilevel optimization problem, comprising an inner loop that simulates the victim's finetuning and an outer loop that optimizes the backdoor properties based on this simulation.

Inner Optimization The purpose of the inner optimization is to forecast the state of the model parameters after the victim performs finetuning. Since the attacker lacks access to the victim's private dataset $\mathcal{D}_v$, directly simulating this process is infeasible. Our key insight is to approximate this process by simulating a k-step gradient descent update using the attacker's own proxy dataset $\mathcal{D}_a$, which provides a first-order approximation of the parameter update *direction*.

Let the initial poisoned model parameters be θ_p. We denote the parameters at the beginning of the inner loop simulation as $\theta_{p,0} \equiv \theta_p$. The inner loop then iteratively computes a sequence of hypothetical parameter updates for $i = 1, \ldots, k$. At each step, we sample a mini-batch (x, y) from $\mathcal{D}_a$ and compute a standard finetuning loss:

$$\mathcal{L}_f(x, y; \theta_{p,i-1}) = \mathcal{L}(M(x; \theta_{p,i-1}), y), \tag{1}$$

where $\mathcal{L}$ denotes the cross-entropy loss and M is the model's output given parameters from the previous step. The parameters are then updated according to a gradient descent rule:

$$\theta_{p,i} = \theta_{p,i-1} - \alpha \nabla_\theta \mathcal{L}_f(x, y; \theta_{p,i-1}), \quad (2)$$

where α is a simulated learning rate that represents the anticipated magnitude of the victim's updates. After k iterations, the simulation yields the final anticipated parameters $\theta'_p \equiv \theta_{p,k}$. Note that θ'_p is not an independent set of parameters, but a differentiable function of the initial parameters θ_p obtained by composing k gradient update steps. This end-to-end differentiability is essential for our bilevel approach, as it allows the outer loop objective to backpropagate *through* the simulated finetuning process and directly optimize θ_p for malicious behavior in its future, post-finetuning state θ'_p.

Outer Optimization The outer optimization leverages the simulated post-finetuning parameters θ'_p to jointly optimize the initial poisoned parameters θ_p and the trigger pattern δ. It is driven by a composite objective that balances three goals: lethality, utility, and stealth.

First, to encode malicious functionality, the **lethality loss** $\mathcal{L}_{\text{lethality}}$ is defined to activate the backdoor only in the post-finetuning state. It is computed using the simulated future parameters θ'_p and enforces that any input embedded with the trigger δ is classified into the target class y_t. Let $x + \delta$ be the triggered input, with the trigger constrained to be imperceptible by $\|\delta\|_\infty \leq \epsilon$. The lethal loss is defined as

$$\mathcal{L}_{\text{lethality}} = \mathcal{L}(M(x + \delta; \theta'_p), y_t), \quad \text{s.t. } \|\delta\|_\infty \leq \epsilon. \quad (3)$$

This term encourages the finetuning trajectory to converge to a state where the backdoor is effective.

Second, to ensure that the compromised model remains useful to the downstream user, its performance on benign inputs should be preserved after finetuning. The **utility loss** $\mathcal{L}_{\text{utility}}$ penalizes degradation in performance on clean data. Similar to the lethality loss, it is computed using the simulated future parameters θ'_p, but on untriggered data from the proxy dataset. Its formulation is for the legitimate task:

$$\mathcal{L}_{\text{utility}} = \mathcal{L}(M(x; \theta'_p), y). \quad (4)$$

By minimizing this loss, the attacker encourages the model to maintain high clean accuracy after finetuning, so that the backdoor does not come at the cost of noticeable utility loss.

Third, while the lethality and utility losses shape the model's future behavior, achieving this is futile if the model is flagged by security audits prior to its release. A model that is merely accurate but exhibits unusual output patterns can be easily identified as anomalous. Therefore, to ensure the backdoor remains dormant and undetectable, we must enforce a much stronger condition of behavioral indistinguishability. For this, we draw inspiration from the principles of knowledge distillation (Hinton, Vinyals, and Dean 2015). We introduce a **stealth loss**, $\mathcal{L}_{\text{stealth}}$, by utilizing the original benign model as a teacher and the trainable backdoored model as a student, to align the output distributions of the benign and backdoored models. This loss operates on the current, pre-finetuning parameters θ_p and compels the poisoned model to mimic the clean teacher model's behavior. This behavioral cloning is achieved by minimizing the Kullback-Leibler (KL) divergence between the softmax output probability distributions of the frozen, clean reference model, $M(\cdot; \theta_c)$, and our trainable poisoned model, $M(\cdot; \theta_p)$. The stealth loss is defined as:

$$\mathcal{L}_{\text{stealth}} = D_{\text{KL}}(P(x; \theta_c) \parallel P(x; \theta_p)) \quad (5)$$

where $P(x; \theta) = \text{softmax}(M(x; \theta))$ denotes the model's predicted class probability distribution for input x. Minimizing this KL divergence compels the student (poisoned) model to align not only with the teacher's (clean) hard predictions but also with its soft confidence scores and the inter-class relationships of the teacher model, making the dormant backdoor difficult to distinguish via standard analyses.

These three objectives are combined into a single composite loss function, $\mathcal{L}_{\text{total}}$, defined as:

$$\mathcal{L}_{\text{total}} = \mathcal{L}_{\text{lethality}} + \lambda_u \mathcal{L}_{\text{utility}} + \lambda_s \mathcal{L}_{\text{stealth}} \quad (6)$$

where λ_u and λ_s control the contributions of the utility and stealth terms. Through standard gradient-based optimization, the initial model parameters θ_p and the trigger pattern δ are jointly updated according to the gradients of $\mathcal{L}_{\text{total}}$. This process iteratively refines θ_p and δ toward a configuration that (i) yields a strong backdoor after finetuning, (ii) preserves clean accuracy for downstream tasks, and (iii) remains stealthy before deployment.

Algorithm 1: Dormant Backdoor Training Pipeline

Require: Clean model parameters θ_c, proxy dataset D_a. target class y_t, Hyperparameters: $\eta_p, \eta_\delta, \lambda_u, \lambda_s, E, k, \alpha$.
Ensure: Poisoned model parameters θ_p and trigger δ.

$\theta_p \leftarrow \theta_c$
Initialize δ randomly.
for epoch = 1 to E **do**
  for each $(x, y) \in D_a$ **do**
    ▷ Inner optimization to find parameters θ'_p
    $\theta_{p,0} \leftarrow \theta_p$
    for $i = 1$ to k **do**
      $\theta_{p,i} \leftarrow \theta_{p,i-1} - \alpha \nabla_\theta \mathcal{L}(M(x; \theta_{p,i-1}), y)$
    end for
    $\theta'_p \leftarrow \theta_{p,k}$
    ▷ Outer optimization objective
    $\mathcal{L}_{\text{total}} \leftarrow \mathcal{L}(M(x + \delta; \theta'_p), y_t) + \lambda_u \mathcal{L}(M(x; \theta'_p), y) + \lambda_s D_{\text{KL}}(P(x; \theta_c) \parallel P(x; \theta_p))$
    ▷ Joint optimization step
    $\theta_p \leftarrow \theta_p - \eta_p \nabla_{\theta_p} \mathcal{L}_{\text{total}}$
    $\delta \leftarrow \delta - \eta_\delta \nabla_\delta \mathcal{L}_{\text{total}}$
  end for
end for
return θ_p, δ

Experiments

Datasets and Experiment Settings

We use ImageNet (Deng et al. 2009) as the attacker's proxy dataset and evaluate performance on five downstream datasets: Caltech101 (Fei-Fei, Fergus, and Perona 2004), OxfordPets (Parkhi et al. 2012), Flowers102 (Nilsback and Zisserman 2008), Food101 (Bossard, Guillaumin,

Dataset	Method	Scenario 1: Unseen Classes			Scenario 2: Unseen Datasets			Harmonic Mean		
		CA	ASR_b	ASR_a	CA	ASR_b	ASR_a	CA	ASR_b	ASR_a
Caltech101	Clean	93.92	-	-	96.26	-	-	95.07	-	-
	BadNets	92.75	96.43	78.52	93.43	91.16	4.49	93.09	93.74	8.49
	Blended	93.23	97.53	83.42	93.85	99.96	28.72	93.54	98.74	42.66
	BadCLIP	93.71	98.99	98.91	94.95	98.52	98.13	94.33	98.75	98.52
	Ours	93.83	0.28	**99.52**	95.64	0.13	**99.16**	94.73	0.18	**99.34**
Flower102	Clean	79.82	-	-	90.63	-	-	84.86	-	-
	BadNets	74.11	83.45	68.97	86.58	99.51	20.58	79.86	90.79	31.69
	Blended	75.40	84.58	72.66	87.46	99.89	32.89	80.99	91.59	45.24
	BadCLIP	74.53	99.63	91.42	87.99	99.96	85.25	80.69	99.79	88.23
	Ours	77.95	0.24	**99.25**	88.86	0.11	**99.19**	83.03	0.15	**99.22**
Food101	Clean	82.74	-	-	84.88	-	-	83.79	-	-
	BadNets	79.44	99.90	98.62	81.62	99.83	40.15	80.52	99.86	57.07
	Blended	80.68	99.98	99.12	81.83	98.75	44.80	81.25	99.36	61.71
	BadCLIP	81.32	99.37	98.63	85.17	99.21	89.62	83.20	99.29	93.92
	Ours	81.92	0.74	**99.82**	85.23	0.61	**99.63**	83.54	0.67	**99.72**
OxfordPets	Clean	91.03	-	-	92.36	-	-	91.69	-	-
	BadNets	89.32	98.94	78.79	91.43	99.40	56.98	90.36	99.17	66.24
	Blended	90.27	99.87	80.70	91.76	99.02	86.15	91.01	99.44	83.35
	BadCLIP	91.87	98.82	80.20	91.80	98.84	91.18	91.83	98.83	85.27
	Ours	91.32	0.92	**99.98**	92.03	0.12	**99.58**	91.67	0.21	**99.78**
SUN397	Clean	73.43	-	-	75.56	-	-	74.48	-	-
	BadNets	71.72	99.80	82.91	73.24	97.92	63.58	72.47	98.85	72.01
	Blended	71.79	99.96	84.93	73.16	99.18	64.57	72.47	99.57	73.37
	BadCLIP	72.83	99.31	68.30	76.79	99.89	74.21	74.76	99.60	71.16
	Ours	73.27	0.35	**99.93**	76.71	0.24	**99.67**	74.95	0.28	**99.80**

Table 1: Comprehensive performance evaluation of our data-agnostic attack against baseline methods across two challenging scenarios: unseen classes and unseen datasets. All experiments are conducted on the ViT-B/16 model, finetuned with LoRA, and the results are averaged over three runs. **CA (%)**: clean accuracy on the downstream task. **ASR_b (%)**: attack success rate **before** finetuning, indicating the backdoor's initial stealthiness (lower is better). **ASR_a (%)**: attack success rate **after** finetuning, indicating the backdoor's lethality (higher is better). Best ASR_a results are highlighted in bold. Our method consistently achieves high post-finetuning lethality while remaining dormant before finetuning.

and Van Gool 2014), and SUN397 (Xiao et al. 2010). Our evaluation proceeds in two settings. For unseen classes generalization, following BadCLIP (Bai et al. 2024), we split each dataset's classes into two disjoint halves, using one for attack generation and the other for finetuning and testing. For unseen datasets generalization, we generate the backdoor using ImageNet as the only proxy data, and the victim then finetunes and evaluates on each of the five downstream datasets. This setting simulates a more realistic model hub scenario where the attacker has no prior knowledge of the downstream tasks or datasets.

Baseline Attacks and Defense Benchmarks

To situate *Dormant Backdoor* within the existing landscape, we benchmark it against three representative backdoor attacks and seven state-of-the-art defense techniques.

Baseline attacks We benchmark our method against three representative attacks. For classical static-trigger attacks in the image domain, we use BadNet (Gu et al. 2019) and Blended (Chen et al. 2017). For the multimodal setting, we use BadCLIP (Bai et al. 2024), the state-of-the-art attack on vision-language models.

Defense methods To assess attack stealth and resilience, we evaluate against a benchmark of seven state-of-the-art defense methods: Neural Cleanse (Wang et al. 2019), STRIP (Gao et al. 2019), GangSweep (Zhu et al. 2020), DLTND and DF-TND (Wang et al. 2020), Causality-Inspired Backdoor Defense (CBD)(Zhang et al. 2023), and CleanCLIP(Bansal et al. 2023). This comprehensive suite allows us to rigorously test both the attack's initial stealth and its persistence after finetuning.

Implementation Details

Unless otherwise specified, we use CLIP with a ViT-B/16 image encoder as our backbone and adopt a 16-shot setting for the downstream user. Dormant Backdoor is trained in the bilevel optimization framework described in the Method section. We use the Adam optimizer for both the model parameters and the trigger, and optimize a weighted sum of lethality, utility, and stealth losses. The weight coefficients are set to $\lambda_s = 15.0$ for the stealth loss and $\lambda_u = 1.5$ for the utility loss. The trigger perturbation is constrained in ℓ_∞-norm with a maximum magnitude $\epsilon = 10/255$, and the first class of each dataset is used as the target class.

For the victim's finetuning, we adopt a standard PEFT setting and use LoRA as the default adaptation method. Unless otherwise stated, the victim finetunes the poisoned CLIP model using AdamW with a cosine annealing learning rate scheduler. All reported results are averaged over three independent runs with different random seeds.

Evaluation Metrics

We evaluate all methods using three complementary metrics that jointly capture utility, stealth, and lethality. **Clean Accuracy (CA)** is the top-1 classification accuracy on the clean test set after the victim's finetuning, measuring how well the model retains its nominal task performance. **Attack Success Rate before finetuning ($\mathbf{ASR_b}$)** is the percentage of trigger-embedded inputs that are classified into the target class *before* any downstream finetuning. A low $\mathrm{ASR_b}$ indicates that the backdoor is dormant and hard to detect at predeployment time. **Attack Success Rate after finetuning ($\mathbf{ASR_a}$)** measures *after* the victim has finetuned the model. A high $\mathrm{ASR_a}$ confirms the backdoor was successfully activated by the finetuning. To highlight the performance trade-off between the unseen classes and unseen datasets scenarios, we also compute the harmonic mean of the results for each metric, following (Xian, Schiele, and Akata 2017).

Experiment Analysis

Table 1 compares *Dormant Backdoor* with baseline attacks across our two evaluation settings, revealing a clear performance gap. Our attack demonstrates a perfect dormant-to-lethal transition. Before finetuning, its success rate ($\mathrm{ASR_b}$) is near zero, ensuring complete stealth. After finetuning, $\mathrm{ASR_a}$ climbs above 99% on all datasets, confirming successful activation regardless of whether the classes or the entire dataset were unseen. In contrast, the baseline attacks are active from the start (high $\mathrm{ASR_b}$), making them easily detectable, and their effectiveness then weakens during the victim's finetuning. Crucially, this lethality does not compromise utility. The model's clean accuracy (CA) remains within a few percentage points of a benign model across all trials, confirming the effectiveness of our utility-preserving objective. These results provide strong evidence for our central claim: by binding the backdoor to the finetuning process itself, our method achieves near-perfect stealth and reliability, even when the downstream data is entirely unknown to the attacker.

Robustness to Different Finetuning Methods

A practical backdoor must withstand diverse finetuning setups. To stress-test our attack, we begin with a *default configuration*—ViT-B/16, LoRA adaptation, 20 epochs, AdamW, CosineAnnealingLR, and a learning rate of 2e-4, and then vary one hyperparameter at a time while freezing the others. The sweep covers various model scales, five adaptation strategies (from full finetuning to PEFT methods), multiple optimizers, learning rates, and training epochs.

Across this grid, the clean accuracy (CA) of the poisoned model fluctuates within a narrow $\pm 4\%$ band relative to the default, attesting that our utility constraint generalizes well. More importantly, the post-finetuning attack success rate ($\mathrm{ASR_a}$) remains $\geq 94\%$ in *every* setting. Even the most disruptive full finetuning, a ten-times larger learning rate, or switching to prompt tuning, only dent $\mathrm{ASR_a}$ by at most 5%. This confirms that our bilevel optimization successfully exploits universal process level regularities, making the attack resilient to real world finetuning variability rather than relying on a specific setup.

Category	Configuration	CA (%)	$\mathrm{ASR_a}$ (%)
Architecture	ViT-B/32	91.92	99.24
	ViT-B/16	93.83	99.52
	ViT-L/14	95.92	99.93
FT Method	Full-FT	89.2	96.50
	LoRA	93.83	99.52
	BitFit	94.03	95.93
	Prompt Tuning	90.32	94.14
	LayerNorm Tuning	92.45	98.95
Optimizer	SGD	94.07	100.0
	Adam	92.97	99.43
	AdamW	93.83	99.52
LR Scheduler	Constant LR	93.46	99.57
	StepLR	92.26	99.89
	CosineAnnealingLR	93.83	99.52
Learning Rate	2e-3	89.96	95.72
	2e-4	93.83	99.52
	2e-5	92.01	99.23
Epochs	**20**	93.83	99.52
	50	96.32	99.10

Table 2: Robustness Evaluation Against Various Finetuning Configurations. In each subsequent section, we vary a single configuration dimension (**default settings are bolded**) while keeping others fixed. The consistently high $\mathrm{ASR_a}$ across all settings highlights the attack's extreme robustness.

Ablation Study

We conduct an ablation study to understand the role of different components in Dormant Backdoor. Table 3 reports results on Caltech101 in the unseen classes scenario under the default finetuning configuration. The full method achieves high CA, near-zero $\mathrm{ASR_b}$, and high $\mathrm{ASR_a}$, indicating that it simultaneously satisfies utility, stealth, and lethality.

Method Setting	CA (%)	$\mathrm{ASR_b}$ (%)	$\mathrm{ASR_a}$ (%)
Full Method	**93.83**	**0.28**	**99.52**
W/o Bilevel Optimization	89.12	84.95	94.32
W/o Stealth Loss ($\mathcal{L}_s$)	93.02	99.21	99.98
W/o Utility Loss ($\mathcal{L}_u$)	52.43	0.19	100.0

Table 3: Ablation study of core components. We analyze the impact of each component by selectively disabling. The experiments are conducted on the Caltech101 dataset with the default finetuning configuration (ViT-B/16, LoRA). Only our full method can simultaneously achieve high utility (CA), stealth (low $\mathrm{ASR_b}$), and lethality (high $\mathrm{ASR_a}$).

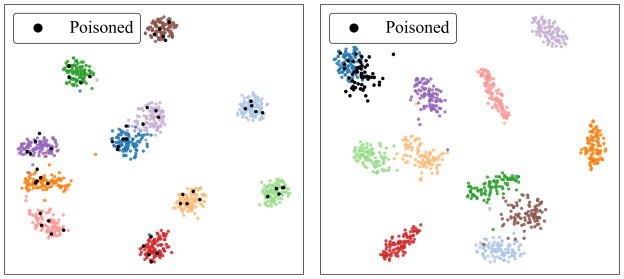
Removing any single component leads to a critical failure. Disabling *bilevel optimization* prevents the model from anticipating the finetuning trajectory, causing pre-deployment stealth to collapse. Similarly, dropping the *stealth loss* $\mathcal{L}_s$ eliminates behavioral mimicry, making the trigger immediately active and exposing the backdoor. Finally, omitting the *utility loss* $\mathcal{L}_u$ decimates the model's clean accuracy, rendering it useless. Taken together, these results confirm that bilevel optimization forecasts the victim's update path, $\mathcal{L}_s$ suppresses premature activation, and $\mathcal{L}_u$ preserves task performance; only their concerted action realizes a backdoor that is *dormant before finetuning yet deadly afterwards*.

Visualizations

Visualization of Backdoor Activation To gain intuition about how Dormant Backdoor behaves before and after finetuning, we visualize the model's output logits using t-SNE (van der Maaten and Hinton 2008). We randomly sample images from 10 classes in the Caltech101 test set, construct their triggered counterparts, and project the logits of the poisoned model into two dimensions.

As shown in Figure 3a, before finetuning, the poisoned samples (black points) remain near their original class clusters. This visualizes the attack's stealth: the model perceives a triggered input as similar to its clean version, keeping the backdoor dormant. Figure 3b reveals a dramatic transformation after finetuning. The process acts as an activation trigger, causing the logits of all poisoned samples to detach from their original classes and converge into a single, tight cluster. This provides compelling visual evidence that the backdoor is unequivocally awakened by the finetuning process itself, validating our core hypothesis.

Visualization of Triggered Samples Figure 4 pairs five benign images (top row) with their corresponding poisoned counterparts (bottom row). The triggered images are visually almost indistinguishable from their original, benign counterparts. This demonstrates the highly stealthy and imperceptible nature of the applied trigger.



(a) Before Finetuning (b) After Finetuning

Figure 3: t-SNE of model logits before and after finetuning. Colored points are benign classes; black points are poisoned samples. (a) Before finetuning, poisons align with original clusters (dormant). (b) After finetuning, they collapse into the target cluster (activated).

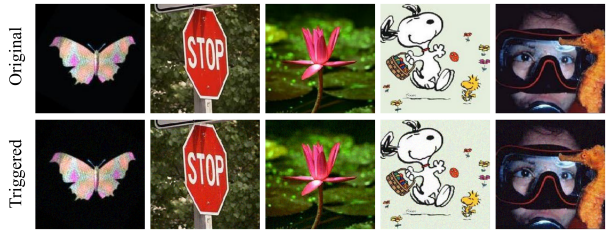


Figure 4: Illustration of benign and triggered images.

Resistance to Backdoor Detection

A practical attack must evade strong defenses. Table 4 shows the performance of four attacks against seven detection methods: **NC** (Neural Cleanse), **ST** (STRIP), **GS** (GangSweep), **DL** (data limited TND-DL), **DF** (data free TND-DF), **CBD** (Causality-Inspired Backdoor Defense), and **CC** (CleanCLIP). A ✘ denotes the detector flagged the model as malicious; a ✓ denotes it passed inspection.

Attack	NC	ST	GS	DL	DF	CBD	CC
BadNets	✘	✘	✘	✘	✘	✘	✘
Blend	✘	✘	✘	✘	✘	✘	✘
BadCLIP	✓	✓	✘	✘	✓	✓	✘
Ours	✓	✓	✓	✓	✓	✓	✓

Table 4: Evaluation of attack stealth against seven state-of-the-art backdoor defenses. A ✓ indicates that the attack successfully evades the detector, while ✘ indicates that it is detected. Dormant Backdoor evades all detectors, demonstrating a significant advantage in stealth over baseline methods.

Classical attacks like BadNets and Blended are caught by *every* defense, as their static triggers create conspicuous statistical anomalies. BadCLIP fares slightly better but is still detected by several methods that identify abnormal activation paths. Our attack, in stark contrast, evades all seven defenses. Existing defenses fundamentally assume the backdoor is already active and searchable before finetuning. Our dormant backdoor violates this premise, as the malicious behavior only materializes after the victim's own finetuning process, rendering these audit tools ineffective. This confirms that by tying activation to the finetuning process, our attack bypasses the entire generation of backdoor defenses.

Conclusion

We presented Dormant Backdoor, a backdoor attack that is activated during model finetuning and operates in a data-agnostic setting without access to downstream tasks or data. By exploiting the optimization process itself as a process-as-trigger mechanism, the attack remains dormant before deployment, evades a wide range of existing backdoor defenses, and achieves near-perfect attack success rates after finetuning while preserving clean accuracy. Our findings expose a previously underexplored vulnerability in model supply chains and motivate the design of defenses that explicitly monitor and secure the model adaptation process.

Acknowledgments

This work is supported by the Talent Fund of Beijing Jiaotong University (No.2024XKRC047), Beijing NSF (No.L242021) and the Open Project of Anhui Provincial Key Laboratory of Multimodal Cognitive Computation, Anhui University (No. MMC202406).

References

Bai, J.; Gao, K.; Min, S.; Xia, S.-T.; Li, Z.; and Liu, W. 2024. BadCLIP: Trigger-Aware Prompt Learning for Backdoor Attacks on CLIP. In *CVPR*.

Bansal, H.; Singhi, N.; Yang, Y.; Yin, F.; Grover, A.; and Chang, K.-W. 2023. CleanCLIP: Mitigating Data Poisoning Attacks in Multimodal Contrastive Learning. arXiv:2303.03323.

Barni, M.; Kallas, K.; and Tondi, B. 2019. A new Backdoor Attack in CNNs by training set corruption without label poisoning. arXiv:1902.11237.

Bossard, L.; Guillaumin, M.; and Van Gool, L. 2014. Food-101 – Mining Discriminative Components with Random Forests. In *European Conference on Computer Vision*.

Chen, X.; Liu, C.; Li, B.; Lu, K.; and Song, D. 2017. Targeted Backdoor Attacks on Deep Learning Systems Using Data Poisoning. arXiv:1712.05526.

Deng, J.; Dong, W.; Socher, R.; Li, L.-J.; Li, K.; and Fei-Fei, L. 2009. ImageNet: A large-scale hierarchical image database. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, 248–255.

Fei-Fei, L.; Fergus, R.; and Perona, P. 2004. Learning Generative Visual Models from Few Training Examples: An Incremental Bayesian Approach Tested on 101 Object Categories. In *2004 Conference on Computer Vision and Pattern Recognition Workshop*, 178–178.

Gao, Y.; Xu, C.; Wang, D.; Chen, S.; Ranasinghe, D. C.; and Nepal, S. 2019. Strip: A defence against trojan attacks on deep neural networks. In *Proceedings of the 35th annual computer security applications conference*, 113–125.

Gu, T.; Dolan-Gavitt, B.; and Garg, S. 2019. BadNets: Identifying Vulnerabilities in the Machine Learning Model Supply Chain. arXiv:1708.06733.

Gu, T.; Liu, K.; Dolan-Gavitt, B.; and Garg, S. 2019. BadNets: Evaluating Backdooring Attacks on Deep Neural Networks. *IEEE Access*, 7: 47230–47244.

Hinton, G.; Vinyals, O.; and Dean, J. 2015. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*.

Hu, E. J.; Shen, Y.; Wallis, P.; Allen-Zhu, Z.; Li, Y.; Wang, S.; Wang, L.; and Chen, W. 2021. LoRA: Low-Rank Adaptation of Large Language Models. arXiv:2106.09685.

Li, Z.; Su, Y.; and Collier, N. 2025. A Survey on Prompt Tuning. arXiv:2507.06085.

Li, Z.; Sun, H.; Xia, P.; Xia, B.; Rui, X.; Zhang, W.; Guo, Q.; Fu, Z.; and Li, B. 2024. A Proxy Attack-Free Strategy for Practically Improving the Poisoning Efficiency in Backdoor Attacks. *IEEE Transactions on Information Forensics and Security*, 19: 9730–9743.

Li, Z.; Xia, P.; Sun, H.; Zeng, Y.; Zhang, W.; and Li, B. 2025. Explore the Effect of Data Selection on Poison Efficiency in Backdoor Attacks. *IEEE Transactions on Dependable and Secure Computing*, 1–18.

Liu, K.; Dolan-Gavitt, B.; and Garg, S. 2018. Fine-Pruning: Defending Against Backdooring Attacks on Deep Neural Networks. arXiv:1805.12185.

Liu, Y.; Ma, S.; Aafer, Y.; Lee, W.-C.; Zhai, J.; Wang, W.; and Zhang, X. 2018. Trojaning Attack on Neural Networks. In *25th Annual Network and Distributed System Security Symposium, NDSS 2018, San Diego, California, USA, February 18-221, 2018*. The Internet Society.

Nilsback, M.-E.; and Zisserman, A. 2008. Automated Flower Classification over a Large Number of Classes. In *2008 Sixth Indian Conference on Computer Vision, Graphics & Image Processing*, 722–729.

Ning, R.; Li, J.; Xin, C.; Wu, H.; and Wang, C. 2022. Hibernated Backdoor: A Mutual Information Empowered Backdoor Attack to Deep Neural Networks. *Proceedings of the AAAI Conference on Artificial Intelligence*, 36(9): 10309–10318.

Parkhi, O. M.; Vedaldi, A.; Zisserman, A.; and Jawahar, C. V. 2012. Cats and dogs. In *2012 IEEE Conference on Computer Vision and Pattern Recognition*, 3498–3505.

Sha, Z.; He, X.; Berrang, P.; Humbert, M.; and Zhang, Y. 2022. Fine-Tuning Is All You Need to Mitigate Backdoor Attacks. arXiv:2212.09067.

Turner, A.; Tsipras, D.; and Madry, A. 2019. Clean-Label Backdoor Attacks.

van der Maaten, L.; and Hinton, G. 2008. Visualizing Data using t-SNE. *Journal of Machine Learning Research*, 9(86): 2579–2605.

Wang, B.; Yao, Y.; Shan, S.; Li, H.; Viswanath, B.; Zheng, H.; and Zhao, B. Y. 2019. Neural Cleanse: Identifying and Mitigating Backdoor Attacks in Neural Networks. In *2019 IEEE Symposium on Security and Privacy (SP)*, 707–723.

Wang, R.; Zhang, G.; Liu, S.; Chen, P.-Y.; Xiong, J.; and Wang, M. 2020. Practical Detection of Trojan Neural Networks: Data-Limited and Data-Free Cases. In *Proceedings of the European Conference on Computer Vision (ECCV)*.

Xian, Y.; Schiele, B.; and Akata, Z. 2017. Zero-Shot Learning — The Good, the Bad and the Ugly. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 3077–3086.

Xiao, J.; Hays, J.; Ehinger, K. A.; Oliva, A.; and Torralba, A. 2010. SUN database: Large-scale scene recognition from abbey to zoo. In *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 3485–3492.

Yao, Y.; Li, H.; Zheng, H.; and Zhao, B. Y. 2019. Latent Backdoor Attacks on Deep Neural Networks. In *Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security*, CCS '19, 2041–2055. New York, NY, USA: Association for Computing Machinery. ISBN 9781450367479.

Zaken, E. B.; Ravfogel, S.; and Goldberg, Y. 2022. BitFit: Simple Parameter-efficient Fine-tuning for Transformer-based Masked Language-models. arXiv:2106.10199.

Zeng, Y.; Pan, M.; Just, H. A.; Lyu, L.; Qiu, M.; and Jia, R. 2023. Narcissus: A Practical Clean-Label Backdoor Attack with Limited Information. In *Proceedings of the 2023 ACM SIGSAC Conference on Computer and Communications Security*, CCS '23, 771–785. New York, NY, USA: Association for Computing Machinery. ISBN 9798400700507.

Zhang, Z.; Liu, Q.; Wang, Z.; Lu, Z.; and Hu, Q. 2023. Backdoor Defense via Deconfounded Representation Learning. In *2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 12228–12238.

Zhu, L.; Ning, R.; Wang, C.; Xin, C.; and Wu, H. 2020. GangSweep: Sweep out Neural Backdoors by GAN. In *Proceedings of the 28th ACM International Conference on Multimedia*, MM '20, 3173–3181. New York, NY, USA: Association for Computing Machinery. ISBN 9781450379885.

Zhu, M.; Wei, S.; Shen, L.; Fan, Y.; and Wu, B. 2023. Enhancing Fine-Tuning Based Backdoor Defense with Sharpness-Aware Minimization. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 4466–4477.