Persistent Backdoor Attacks in class-incremental learning via Structural Invariant Anchoring

Anonymous Authors[1]

Abstract

Continual Learning (CL) continually performs parameter updates, posing a significant challenge to backdoor persistence. In this paper, we reveal that the most advanced attack relies on an implicit assumption that task-critical neurons remain stable across task learning; however, it does not hold in class-incremental learning (CIL). This exposes a critical research gap: the backdoor persistence in CIL is still an open question. Inspired by the function stability despite neuron instability, we discover that the CIL models preserve task knowledge in shallow, structurally invariant subspaces. Motivated by the findings, we propose PBTO, the first persistent and targeted backdoor attack in CIL. PBTO trains a surrogate model on proxy tasks to obtain the parameter trajectory. Then, it optimizes a universal trigger that ensures misclassification to the target label across all model states and anchors trigger embeddings in shallow layers. Experimental results verify that PBTO maintains effectiveness even after learning multiple tasks, while existing methods degrade to below 10%.

1. Introduction

Classical machine learning (ML) paradigms train models for a single predefined task, which is inadequate for real-world scenarios that continuously encounter new tasks. For instance, facial recognition systems need to incorporate new individuals, and spam filters must detect emerging threats. To address this challenge, researchers have developed continual learning (CL), a paradigm that enables models to learn new tasks sequentially while preserving prior knowledge. CL is commonly categorized into three settings with varying degrees of task constraints: task-incremental learning (TIL), domain-incremental learning (DIL), and class-incremental learning (CIL). TIL employs task-specific output heads and requires task identity at inference. DIL adopts a single classifier under a shared label space across different tasks. CIL incrementally introduces new classes without task identity or a shared label space, thereby presenting the most challenging and practically continual learning scenario.

Akin to classical ML, CL typically relies on external data for model training. This fundamental dependence on untrusted data exposes CL to backdoor attack threats (Gu et al., 2017; Guo et al., 2025; Jiang et al., 2022). Adversaries inject a small set of poisoned samples into the training dataset. The poisoned samples contain designated triggers and are labeled with an attacker-specified target class, inducing a hidden backdoor association between the triggers and the target label during training. At inference, poisoned inputs stamped with the trigger are classified as the target class, whereas clean inputs remain unaffected. However, directly applying existing backdoor attacks to CL presents significant attack performance degradation (Guo et al., 2025). This is primarily attributed to the dynamic nature of CL, where continuous parameter updates progressively eliminate the established backdoor associations. As a result, backdoor persistence emerges as a fundamental challenge in CL.

Existing backdoor attacks tailored to CL include three strategies, each attempting to address backdoor persistence but exhibiting limitations under CIL settings. Specifically, (1) continuous poisoning-based attack (Gao & Liu, 2025) assumes continuously to inject poisoned samples throughout the CIL training, which is difficult to achieve in realistic deployment setting. (2) TIL-oriented backdoor attacks (Guo et al., 2025) rely on task-specific neurons to preserve backdoor. However, CIL operates within a shared feature space and continuously repurposes existing neurons to accommodate new knowledge, inevitably undermining backdoor associations established under TIL assumptions. (3) CIL-oriented backdoor attack (Jiang et al., 2022) embeds triggers into pre-trained backbones, but is limited to untargeted attacks as target classes emerge incrementally and cannot be pre-specified. Consequently, a research gap persists: no existing work simultaneously achieves persistent and targeted backdoor attacks under one-time poisoning constraints, leaving CIL security against such threats unexplored.

[1]Anonymous Institution, Anonymous City, Anonymous Region, Anonymous Country. Correspondence to: Anonymous Author <anon.email@domain.com>.

Preliminary work. Under review by the International Conference on Machine Learning (ICML). Do not distribute.

In this paper, we investigate the limitations of LTB (Guo et al., 2025), which, to our best knowledge, is the state-of-the-art backdoor attack in TIL. We reveal that LTB's persistence relies on an assumption: task-critical neurons remain invariant across subsequent learning. However, we demonstrated that this assumption does not hold in CIL. Unlike TIL with task-specific architectures, CIL operates under a unified architecture with shared parameters continuously repurposed to adapt to new tasks. Interestingly, CIL models maintain robust performance on previous tasks even under severe neuron-level instability. This suggests that task knowledge is preserved through structural invariants at the subspace level rather than through fixed individual neurons. To verify this, we track the principal subspace of initial task samples and measure its preservation via similarity analysis as new tasks are learned. We discover that shallow layers retain over 80% subspace similarity across tasks, while deeper layers exhibit significant drift. This reveals a critical insight for persistent backdoor design: anchor triggers to structurally invariant shallow subspaces.

Motivated by this insight, we propose Persistent Backdoor Trigger Optimization (PBTO), the first persistent and targeted backdoor attack for CIL. PBTO first simulates CIL parameter evolution by sequentially training a surrogate model on proxy tasks, constructing a model state trajectory that mirrors real-world continual learning dynamics without accessing the victim model. Based on this trajectory, PBTO introduces a dual-objective optimization strategy to design a universal trigger that: (1) misclassifies clean samples to the attacker-specified target label across all surrogate model states, while (2) anchors trigger embeddings to truly invariant structural subspace rather than trajectory-specific artifacts. PBTO achieves this by jointly minimizing trajectory-averaged cross-entropy loss and Gram matrix regularization that aligns trigger representations with target-class samples in shallow layers. To account for backdoor-induced gradient interference, PBTO further employs iterative refinement strategy that alternates between simulating the poisoned trajectory and re-optimizing the trigger.

In summary, our contributions are four-fold. (1) We reveal a critical research gap: no existing work simultaneously achieves persistent and targeted backdoor attacks in CIL, leaving CIL robustness against such threats largely unexplored. (2) We discover that CIL models preserve task knowledge through structurally invariant subspaces in shallow layers. (3) Based on our findings, we propose PBTO, the first persistent and targeted backdoor attack for CIL systems, which anchors triggers to invariant structural subspaces via trajectory simulation and dual-objective optimization. (4) Experimental results indicate that PBTO achieves superior attack success rates (>90%), significantly outperforming existing attacks (<10%), and maintains backdoor persistence across the entire continual learning process.

2. Related Work

2.1. Class-Incremental Learning

Class-Incremental Learning (CIL) represents the most challenging continual learning setting, where models employ a unified classifier with all tasks sharing the same feature space and linear classifier (Rebuffi et al., 2017; Buzzega et al., 2020). This unified architecture necessitates continuous feature repurposing as the model must continuously reorganize its representations to accommodate new classes. Existing CIL methods mitigate catastrophic forgetting mainly through three mechanisms. (1) Replay-based methods (Rebuffi et al., 2017; Buzzega et al., 2020), which store historical samples or synthesize past data, then mix these old samples with new task data during training. (2) Regularization-based methods (Kirkpatrick et al., 2017; Li & Hoiem, 2017) identify important parameters for old tasks via Fisher Information and penalize their modification when learning new tasks. (3) Prompt-based methods (Wang et al., 2022b;a) freeze the feature extractor to prevent forgetting.

2.2. Backdoor Attacks

Backdoor Attacks in Classical Learning. In backdoor attack, adversaries inject poisoned samples containing designated triggers and labeled as an attacker-specified target class into training set. The trained model learns a hidden trigger-target association. At inference, triggered inputs are misclassified to the target class, while clean inputs remain unaffected. Early studies like BadNets (Gu et al., 2017) utilized visible patch-based triggers (Gu et al., 2017). To enhance stealthiness, subsequent research introduced invisible triggers via additive perturbations (Chen et al., 2017), image warping (Nguyen & Tran, 2021), or frequency-domain injection (Zeng et al., 2021). Based on the adversary's capabilities, attacks are generally categorized into poison-only, training-controlled, and model-modified settings (Li et al., 2022). In this paper, we focus on the practical poison-only setting, where the attacker is restricted to modifying a subset of the training data.

Backdoor Attacks in Continual Learning. The security of CL against backdoor attacks remains largely underexplored. Existing studies either require continuous injection of poisoned samples throughout training (Gao & Liu, 2025), limiting practicality under realistic one-time poisoning constraints, or anchor triggers to architecture-specific components (Jiang et al., 2022) and task-specific neurons (Guo et al., 2025). These approaches are ineffective in CIL, where the unified architecture continuously repurposes shared neurons to accommodate new classes, leading to rapid backdoor degradation. Consequently, no existing work simultaneously achieves both persistence and target specification under realistic one-time poisoning constraints in CIL.

2.3. Backdoor Defenses

Backdoor defenses operate at two levels: (1) model-level defenses, which remove backdoor from trained models via neuron pruning (Liu et al., 2018) or knowledge distillation (Li et al., 2021); and (2) input-level defenses that detect or neutralize triggered samples, including trigger inversion methods (Wang et al., 2019; Xu et al., 2024) and feature suppression techniques (Chen et al., 2025). These defenses generally operate under classic ML learning assumptions. We demonstrate that PBTO evades these defenses by anchoring triggers to structural invariants.

3. Revisiting Backdoor Persistence in CIL

3.1. Threat Model and Problem Formulation

Threat Model. In this paper, we consider the poison-only backdoor attack in class-incremental learning (CIL), which imposes minimal assumptions on the adversary. Specifically, we assume that adversary can only poison the training data of a target task by injecting poisoned samples, without access to the model architectures, parameters, gradients or the specific incremental learning algorithms. Moreover, the adversary has no knowledge of the training distribution of the subsequent tasks to be learned after poisoning. Consistent with prior work, we assume the adversary has access to a small number of publicly available data drawn from the same distribution as the target class.

Attacker Goals. In CIL, continuous parameter updates during incremental learning can erase the implanted backdoor. Thus, the adversary pursues three objectives. (1) Effectiveness: poisoned inputs are consistently misclassified to the attacker-specified target label. (2) Persistence: the backdoor remains effective throughout the continual learning process. Once the backdoor is injected during the learning of a target task, it should persist across all subsequent tasks. Despite continuous parameter updates induced by learning new incremental tasks, the model is expected to consistently misclassify poisoned samples to the target label at every stage of the continual learning process. (3) Transferability: the attacks are effective across various model architectures.

Poison-only Backdoor Attacks in CIL. In CIL, a model is continuously trained to adapt to new tasks. Formally, for a sequence of tasks $\mathcal{T} = \{T_1, T_2, \ldots, T_N\}$, the CIL objective is to achieve high accuracy on the current task T_k (where $k \in \{1, 2, \ldots, N\}$) and all previously learned tasks $\{T_1, \ldots, T_{k-1}\}$. Each task T_k ((where $k \in \{1, 2, \ldots, N\}$)) provides a dataset $\mathcal{D}_k = \{(\boldsymbol{x}_i^{(k)}, y_i^{(k)})\}_{i=1}^{|\mathcal{D}_k|}$. The label space $\mathcal{Y}_k$ of task T_k is disjoint from all other tasks, *i.e.*, $\mathcal{Y}_i \cap \mathcal{Y}_j = \emptyset$ for $i \neq j$. Starting from randomly initialized parameters $\boldsymbol{\theta}_0$, at each step k, the parameters $\boldsymbol{\theta}_{k-1}$ are updated on the dataset $\mathcal{D}_k$, producing a parameter trajectory $\boldsymbol{\theta}_0 \xrightarrow{\mathcal{D}_1} \boldsymbol{\theta}_1 \xrightarrow{\mathcal{D}_2} \boldsymbol{\theta}_2 \xrightarrow{\mathcal{D}_3} \cdots \xrightarrow{\mathcal{D}_N} \boldsymbol{\theta}_N$. The final model $\mathbf{F}_{\boldsymbol{\theta}_N}$ (also denoted as $\mathbf{F}_{\boldsymbol{\theta}}$) performs inference over the cumulative label space $\mathcal{Y} = \bigcup_{j=1}^{N} \mathcal{Y}_j$.

In the poison-only backdoor attack, attackers can only inject poisoned samples into the training set of a specific target task. Formally, for a target task T_k with its training set $\mathcal{D}_k$, the attacker first selects a subset of clean samples $\mathcal{D}_s \subset \mathcal{D}_k$. Each clean sample $(\boldsymbol{x}, y) \in \mathcal{D}_s$ is then transformed into a poisoned sample $(\hat{\boldsymbol{x}}, y_t), \hat{\boldsymbol{x}} = \boldsymbol{x} \oplus \boldsymbol{\delta}$, where $\boldsymbol{\delta}$ is the trigger, and $y_t \in \mathcal{Y}_k$ is the attacker-specified target label. Then, the poisoned sample subset $\mathcal{D}_p$ is constructed: $\mathcal{D}_p = \{(\mathcal{G}(\boldsymbol{x}, \boldsymbol{\delta}), y_t) \mid (\boldsymbol{x}, y) \in \mathcal{D}_s\}$. The final poisoned training set for task T_k is constructed by replacing $\mathcal{D}_s$ with the poisoned samples $\mathcal{D}_p$, yielding the poisoned training set $\hat{\mathcal{D}}_k = (\mathcal{D}_k \setminus \mathcal{D}_s) \cup \mathcal{D}_p$. After training on the poisoned dataset $\hat{\mathcal{D}}_k$, the model persistently behaves as:

$$\hat{\mathbf{F}}_{\boldsymbol{\theta}_j}(\boldsymbol{x}) = y, \quad \hat{\mathbf{F}}_{\boldsymbol{\theta}_j}(\hat{\boldsymbol{x}}) = y_t, \quad \forall j \in \{k, \ldots, N\}, \quad (1)$$

where clean inputs $\boldsymbol{x}$ are correctly classified, and poisoned inputs are $\hat{\boldsymbol{x}}$ consistently misclassified to the target label y_t even as the model updates parameters on subsequent tasks.

3.2. Rethinking the Neuron-Level Stability

To the best of our knowledge, LTB (Guo et al., 2025) currently represents the most advanced persistent backdoor attack for Task-Incremental Learning (TIL). It assumes that task-critical neurons, characterized by high Fisher information, remain isolated and their parameter values stay unchanged during subsequent training. This assumption indeed holds in TIL, where task identities are explicitly provided, allowing task-specific architectures (*e.g.*, separate classifier heads) to preserve isolated parameter subspaces for each task. However, in CIL, task identities are unavailable, and all tasks share a unified model architecture. This architectural constraint fundamentally breaks the neuron isolation assumption in LTB: as new tasks are learned, the shared network continuously repurposes existing neurons to accommodate new knowledge, inevitably overwriting previously critical parameters.

To validate the instability of task-critical neurons in CIL, we conduct an analysis by tracking parameter dynamics throughout continual learning. Specifically, we train a ResNet-18 model on CIFAR-10 in a five-task CIL setting, where each task introduces two new classes sequentially. We identify two sets of neurons: (1) neurons that are critical for Task one, determined by high Fisher information values, and (2) neurons that remain exhibit minimal parameter across all five tasks. We then measure the overlap ratio between these two sets to assess whether task-critical neurons remain stable. As shown in Figure 1, this overlap ratio drops to below 10%, demonstrating a severe divergence between task-criticality and parameter stability under CIL's continu-

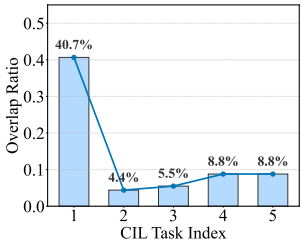


Figure 1. Stability of task-critical neurons in CIL, quantified by the overlap ratio between critical neurons and stable neurons across five tasks.

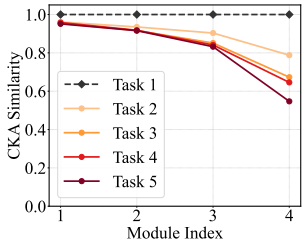


Figure 2. Stability of module-wise feature representations, measured by CKA similarity between Task 1 representations and subsequent tasks.

ous feature repurposing pressure. This finding confirms that the neuron-isolation assumption of LTB fails to hold in CIL.

3.3. From Neuron Isolation to Subspace Anchoring

The analysis in Sec. 3.2 reveals severe instability of individual neurons; however, CIL models maintain robust performance on previously learned tasks, which is CIL's fundamental objective. This observation suggests that task-relevant information may be preserved not solely through fixed neuron values, but potentially via invariant geometric structures in the feature space. This motivates us to shift perspective: instead of tracking individual neurons, we investigate subspace-level invariance across CIL.

To validate the existence of structural invariance, we adopt a natural strategy: tracking the feature subspace across early tasks to observe whether it remains stable after learning new tasks. If an invariant subspace exists, the principal structure captured from the initial task should remain intact during the subsequent tasks' learning. We operationalize this by first extracting the dominant structural representation from Task 1 at a specific layer ℓ, forming feature matrix $\mathbf{\Phi}_\ell^{(1)}$. To establish a reference subspace that captures the most principal directions, we apply Singular Value Decomposition (SVD): $\mathbf{\Phi}_\ell^{(1)} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^\top$. The first r left singular vectors $[\mathbf{u}_1, \ldots, \mathbf{u}_r]$ of $\mathbf{U}_r$, corresponding to the largest singular values, define our r-dimensional principal reference subspace. As training progresses through Tasks 2, 3, $\ldots$, N, we re-extract features for Task 1 samples using the updated model, obtaining $\mathbf{\Phi}_\ell^{(1|t)}$. We measure structural preservation using CKA (Kornblith et al., 2019), which quantifies similarity between representation spaces while remaining invariant to unitary transformations.

We train a ResNet-18 model on CIFAR-10 using a 5-task CIL setting where each task introduces 2 new classes. To analyze representational stability, we extract features from 1,000 Task 1 test samples after each of the four residual blocks (denoted as Modules 1-4, corresponding to network depth) at different training stages. Figure 2 shows that over 50% of Task 1's variance is retained throughout the entire 5-task sequence, confirming the existence of an *Invariant Structal Subspace* that persists despite continuous parameter updates. Meanwhile, this stability is not uniform across depth: shallow layers (Modules 1-2) maintain high CKA similarity across tasks, while the final residual block (Module 4) exhibits significant representational drift.

Insights for Persistent Attack Design. The findings reveal a critical vulnerability in CIL: the structural invariance subspace in shallow layers provides a stable anchor point for backdoor attacks. Since low-level representations remain stable throughout CIL, backdoor triggers embedded within this invariant subspace can persist across tasks. This observation motivates our core attack strategy: anchor backdoor within the invariant subspace of early layers, exploiting the structural stability that enables continual learning to achieve persistent backdoor attack throughout CIL.

4. Persistent Backdoor Trigger Optimization

4.1. Overview

Building on the insights in Sec. 3.3, we propose Persistent Backdoor Trigger Optimization (PBTO), the first persistent backdoor attack designed for the challenging class-incremental learning (CIL) setting. Specifically, PBTO mainly consists of two stages: (1) Trajectory Modeling, where a surrogate model is trained on proxy tasks to simulate the parameter evolution induced by CIL, thereby constructing an approximate feature-space trajectory without access to the victim model during CIL process; and (2) Subspace Anchoring, which optimizes backdoor triggers to induce misclassification of clean inputs to the target label, while simultaneously anchoring the trigger representations in the invariant shallow subspace shared across different model states along the simulated parameter trajectory.

4.2. Trajectory Modeling

The continual parameter evolution in CIL challenges backdoor persistence, as triggers optimized on a single model state are easily disrupted by subsequent feature repurposing. As revealed in Sec. 3.3, shallow feature subspaces exhibit greater stability across CIL tasks. Ideally, a persistent trigger should anchor its representations within such invariant subspaces. However, in poison-only attacks, the victim's subsequent tasks are unavailable. This raises a key question: *how can triggers be anchored to stable subspaces without access to the CIL process?*

To address this challenge, we proactively simulate the parameter evolution process in CIL, thereby approximating the parameter updates trajectory. The key insight is that, in CIL, the parameter updates is largely influenced by the learning mechanism, such as the network architecture and optimiza-

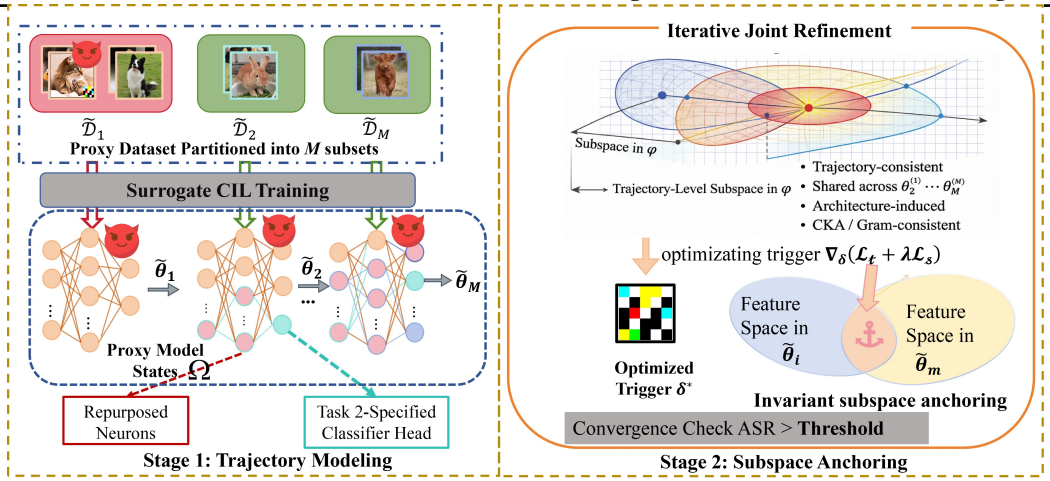


Figure 3. The main pipeline of PBTO. **Stage 1. Trajectory Simulation**: PBTO trains a surrogate model on proxy tasks to simulate CIL parameter evolution, obtaining a trajectory of model states. **Stage 2. Subspace Anchoring**: PBTO optimizes a universal trigger through dual-objective optimization and then iteratively refines the trigger by alternating between trajectory simulation and trigger optimization.

tion dynamics, rather than solely by the specific semantic content of tasks. This property allows us to train a surrogate model on proxy tasks under the CIL settings to capture the parameter update patterns that the victim model is likely to experience. Specifically, we assume the attacker has a local proxy dataset $\tilde{\mathcal{D}}$ that is independently collected from the target class-conditional distribution (*e.g.*, obtainable from publicly available datasets). This assumption aligns with the threat model commonly adopted in existing backdoor attacks, where attackers possess auxiliary data resources. To emulate the continual learning dynamics, we partition the proxy dataset $\tilde{\mathcal{D}}$ into M disjoint subsets:

$$\tilde{\mathcal{D}} = \bigcup_{m=1}^{M} \tilde{\mathcal{D}}_m, \quad \tilde{\mathcal{D}}_i \cap \tilde{\mathcal{D}}_j = \emptyset \text{ for } i \neq j. \tag{2}$$

We then sequentially train a surrogate model on these subsets to simulate M incremental learning phases. This process collects a locally simulated proxy trajectory:

$$\Omega = \left\{ \tilde{\boldsymbol{\theta}}_1, \ldots, \tilde{\boldsymbol{\theta}}_M \right\}. \tag{3}$$

4.3. Trigger Optimization via Subspace Anchoring

The trajectory modeling in Sec. 4.2 yields a sequence of surrogate model states Ω that approximates the victim's parameter evolution. Building on this trajectory, we design a trigger optimization strategy that simultaneously achieves two objectives: (1) trajectory-invariant optimization, which ensures the trigger successfully induces misclassification across all simulated model states; (2) shallow subspace anchoring, which constrains the trigger to the stable shallow subspace. By jointly optimizing these two objectives, the trigger is continually refined and finally anchored to features that remain both effective and stable across all parameter states during the proxy CIL process, thereby achieving persistent backdoor behavior in real CIL scenarios.

Trajectory-Invariant Optimization. We first optimize a universal trigger $\boldsymbol{\delta}$ that maintains effectiveness across all model states in Ω. For a specific task T_k, an attacker selects a subset $\mathcal{D}_s \subset \mathcal{D}_k$ of clean samples for poisoning. For each poisoned input $\hat{\boldsymbol{x}} = \boldsymbol{x} \oplus \boldsymbol{\delta}$ (where $\boldsymbol{x} \in \mathcal{D}_s$), the objective is to consistently redirect the prediction to target label y_t across the entire trajectory. We formulate this as minimizing the averaged cross-entropy loss over all surrogate states:

$$\mathcal{L}_t(\boldsymbol{\delta}) = \frac{1}{|\Omega||\mathcal{D}_s|} \sum_{\tilde{\boldsymbol{\theta}}^{(m)} \in \Omega} \sum_{\boldsymbol{x} \in \mathcal{D}_s} \mathcal{L}_c\left(\tilde{\mathbf{F}}_m(\boldsymbol{x} \oplus \boldsymbol{\delta}), y_t\right), \tag{4}$$

where $\tilde{\mathbf{F}}^{(m)}$ denotes the m-th surrogate model and $\mathcal{L}_c$ is cross-entropy loss function. By optimizing across the entire trajectory rather than a single model snapshot, the trigger is forced to exploit features that persist in CIL.

Shallow Subspace Anchoring. While $\mathcal{L}_t$ enforces backdoor effectiveness across the trajectory, it does not guarantee the trigger exploits the invariant shallow subspace. As a result, the optimization may converge to features stable across proxy models but not inherently invariant, which may fail on unseen tasks. To address this, we introduce a regularization term that explicitly anchors the trigger to stable shallow layers. Inspired by Independent Subspace Analysis (Hyvärinen & Hoyer, 2000), we employ Gram matrices (Gatys et al., 2016) to capture channel-wise correlations in shallow features, encoding feature statistics invariant to spatial configurations. Let $\phi_\ell(\cdot; \tilde{\boldsymbol{\theta}}^{(m)})$ denote the feature extractor up to shallow layer ℓ of surrogate model $\tilde{\mathbf{F}}^{(m)}$ parameterized by $\tilde{\boldsymbol{\theta}}^{(m)}$. We define the anchoring loss as:

$$\mathcal{L}_s(\boldsymbol{\delta}) = \frac{1}{|\Omega||\mathcal{D}_s|} \sum_{\Omega} \sum_{\boldsymbol{x} \in \mathcal{D}_s} \left\| \mathbf{G}_\ell^{(m)}(\boldsymbol{x} \oplus \boldsymbol{\delta}) - \mathbf{G}_\ell^{(m)}(\boldsymbol{r}) \right\|_F^2, \tag{5}$$

where $\mathbf{G}_\ell^{(m)}(\cdot) = \mathbf{G}(\phi_\ell(\cdot; \tilde{\boldsymbol{\theta}}^{(m)}))$ denotes the Gram matrix of shallow layer ℓ, and $\boldsymbol{r}$ is a reference sample from the

target class y_t in the proxy dataset $\tilde{\mathcal{D}}$. Typically, we use ResNet-18 and extract features from the penultimate residual block, which our CKA analysis identified as exhibiting high cross-task stability. By minimizing this loss across the entire trajectory, the trigger is constrained to produce shallow feature correlations similar to genuine target-class samples, thereby anchoring to the stable subspace.

Iterative Joint Refinement. The above trigger optimization is performed on the backdoor-free surrogate models. However, in reality, once the attacker injects the backdoor at task T_k, all subsequent tasks $\{k+1, \ldots, N\}$ are learned on top of the already poisoned model. This creates a trajectory shift: the trigger is optimized on a clean trajectory, but the actual attack unfolds along a poisoned trajectory where backdoor-induced gradient interference alters subsequent learning dynamics. To address this mismatch, we propose an iterative refinement strategy that alternates between trigger optimization and surrogate model training. The complete process is described as follows:

Initialization ($t = 0$). Train M backdoor-free surrogate models sequentially on clean proxy datasets $\{\tilde{\mathcal{D}}_1, \ldots, \tilde{\mathcal{D}}_M\}$, yielding the initial clean trajectory $\Omega^{(0)} = \{\tilde{\boldsymbol{\theta}}_1^{(0)}, \ldots, \tilde{\boldsymbol{\theta}}_M^{(0)}\}$. Optimize the initial trigger:

$$\boldsymbol{\delta}^{(0)} = \arg\min_{\boldsymbol{\delta}} \left[\mathcal{L}_t(\boldsymbol{\delta}) + \lambda \mathcal{L}_s(\boldsymbol{\delta})\right], \tag{6}$$

where $\lambda > 0$ is a hyperparameter balancing the two losses.

Iterative Refinement ($t \geq 1$). For each iteration t, perform two alternating steps:

Step 1: Trajectory Simulation. Using trigger $\boldsymbol{\delta}^{(t-1)}$, construct the poisoned dataset for task T_k:

$$\tilde{\mathcal{D}}_k^{(t)} = \{(\boldsymbol{x} \oplus \boldsymbol{\delta}^{(t-1)}, y_t) \mid \boldsymbol{x} \in \mathcal{D}_s\} \cup (\mathcal{D}_k \setminus \mathcal{D}_s). \tag{7}$$

Simulate the CIL process: train sequentially on tasks $\{T_1, \ldots, T_{k-1}\}$ with clean data, task T_k with poisoned data $\tilde{\mathcal{D}}_k^{(t)}$, and tasks $\{T_{k+1}, \ldots, T_M\}$ with clean data, yielding the poisoned trajectory:

$$\Omega^{(t)} = \{\tilde{\boldsymbol{\theta}}_1^{(t)}, \ldots, \tilde{\boldsymbol{\theta}}_M^{(t)}\}. \tag{8}$$

Step 2: Trigger Re-optimization. Fix trajectory $\Omega^{(t)}$ and update the trigger $\boldsymbol{\delta}^{(t)}$ same as in Eq. 6.

Step 3: Convergence Check. After obtaining $\boldsymbol{\delta}^{(t)}$, compute the attack success rate (ASR) across all proxy tasks:

$$\text{ASR}^{(t)} = \frac{1}{M \cdot |\mathcal{D}_s|} \sum_{m=1}^{M} \sum_{\boldsymbol{x} \in \mathcal{D}_s} \mathbb{I}\left[\tilde{\mathbf{F}}^{(m)}(\boldsymbol{x} \oplus \boldsymbol{\delta}^{(t)}) = y_t\right], \tag{9}$$

where $\tau \in (0, 1]$ is ASR threshold, $\epsilon > 0$ is the convergence tolerance, $\mathbb{I}[\cdot]$ is the indicator function, and $T_{\max}$ is the maximum number of iterations. By iteratively coupling trajectory simulation and trigger optimization, the final trigger $\boldsymbol{\delta}^*$ adapts to the poisoned training updates.

After obtaining the optimized trigger $\boldsymbol{\delta}^*$, the attacker constructs a poisoned training set by injecting the trigger into a small subset of clean samples from the target task. When the victim model is trained on this poisoned dataset, the backdoor becomes embedded into the model's structural invariants. As a consequence, the malicious backdoor behavior remains effective throughout the continual learning process, posing a persistent threat to CIL security.

4.4. Experimental Setup

Datasets and Benchmarks. We evaluate PBTO on CIFAR-10, CIFAR-100, and Tiny-ImageNet. Following standard protocols (Rebuffi et al., 2017), classes are partitioned into N disjoint tasks. Results are reported as mean $\pm$ standard deviation averaged over 3 independent runs with distinct random seeds to cover variations in task ordering and initialization.

Victim Model & Protocol. We adopt iCaRL (Rebuffi et al., 2017) with a ResNet-18 backbone and an episodic buffer of $\mathcal{M} = 2,000$. The attack is injected exclusively in Task 1, representing the "Worst-Case Retention" scenario where the trigger must survive the longest sequence of forgetting (Task $2 \rightarrow N$).

Baselines. We compare against: (1) *Static:* BadNets (Gu et al., 2017) and Blend (Chen et al., 2017); (2) *Stealthy:* WaNet (Nguyen & Tran, 2021) (Warping-based) and LIRA (Doan et al., 2021); and (3) *Adapted SOTA:* LTB (Guo et al., 2025). Since LTB is originally designed for Task-Incremental Learning (TIL), we adapt it to CIL by replacing its multi-head architecture with a unified classifier while retaining its Fisher-based anchoring mechanism.

Implementation Details. To ensure stealthiness, we constrain the L_∞ budget to $\epsilon = 8/255$. We employ a conservative poisoning rate of $\rho = 0.05$ (only 5% samples in Task 1). Triggers are optimized via PGD with $\lambda = 1.0$. All baselines are reproduced using official configurations. Proxy dataset construction is detailed in **Appendix B**.

4.5. Main Results

Table 1 presents the comparative performance on the final model after learning all tasks.

Failure of Semantic Anchoring. The adapted SOTA method, LTB, fails to establish persistence in the CIL setting, as evidenced by the low final ASR on CIFAR-100 (4.7%). This empirical evidence supports our analysis: Fisher Information only captures local stability within the current task's loss landscape. In CIL, the aggressive feature repurposing

Table 1. Performance comparison on the final model after learning all N tasks. Results are averaged over 3 runs (± Std. Dev.).

Method	CIFAR-10		CIFAR-100		Tiny-ImageNet	
	ASR (↑)	BA	ASR (↑)	BA	ASR (↑)	BA
BadNets	12.4±1.5	82.3±0.4	4.5±0.5	68.5±0.7	2.1±0.3	58.4±0.9
WaNet	8.5±2.1	81.9±0.5	3.2±1.1	68.1±0.6	1.5±0.4	57.9±0.8
DRUPE	28.5±3.2	82.0±0.6	14.2±1.9	67.9±0.8	5.6±1.1	58.0±1.0
LTB	12.4±3.5	81.8±0.5	4.7±3.2	67.5±1.2	8.9±2.4	57.8±1.4
PBTO (Ours)	**97.1**±0.4	81.6±0.3	**86.5**±0.9	67.2±0.6	**83.8**±1.1	57.5±1.7

required for new classes inevitably overwrites these "locally stable" semantic neurons, rendering LTB ineffective. Similarly, stealthy baselines like WaNet and DRUPE suffer significant degradation. While WaNet relies on spatial warping, DRUPE enforces strict distribution-preserving constraints. However, these delicate patterns lack structural robustness and are easily washed away by the continuous gradient updates. PBTO achieves superior persistence across all benchmarks. Notably, on the complex Tiny-ImageNet dataset, PBTO maintains a final ASR of 83.8%. This consistency confirms that PBTO successfully decouples the trigger from the volatile semantic subspace, anchoring it instead to robust operational pathways that are preserved by the model for their structural utility. A visual comparison of trigger stealthiness is provided in **Appendix C**.

4.6. Universal Robustness

We conduct strict transferability evaluations on CIFAR-10 to assess whether PBTO exploits generic inductive biases.

Cross-Architecture Transferability. As shown in Table 2, triggers optimized on a surrogate ResNet-18 transfer remarkably well to VGG-16 (78.1% ASR) and DenseNet-121 (82.2% ASR). This indicates that PBTO captures fundamental convolutional features rather than overfitting to specific model parameters.

Table 2. **Cross-Architecture Transferability (CIFAR-10).** Triggers optimized on ResNet-18 achieve high ASR on disjoint architectures.

Source Model	Target Model	LTB	PBTO
ResNet-18	VGG-16	5.5%	**78.1%**
	DenseNet-121	6.3%	**82.2%**
	MobileNetV2	5.8%	**81.5%**

Robustness Across CL Paradigms. We challenge PBTO with diverse CL strategies in Table 3. While LTB collapses under the strict parameter constraints of Regularization (EWC) and the aggressive updates of Replay (DER++), PBTO maintains a robust ASR exceeding 85%. This resilience stems from the fact that PBTO aligns the backdoor with high-utility structural filters. For instance, EWC's penalty on high-Fisher parameters inadvertently *protects* our structural trigger, effectively treating the backdoor as essential prior knowledge.

Table 3. Robustness across CL Algorithms. PBTO maintains high ASR under both Regularization and Replay strategies.

CL Strategy	Method	LTB (ASR)	PBTO (ASR)
Regularization	EWC	8.2%	**88.1%**
Replay	iCaRL	4.7%	**86.5%**
Replay (Strong)	DER++	5.4%	**85.8%**

4.7. Resilience against Defenses

We subject PBTO to a rigorous stress test against state-of-the-art defenses (Table 4). **Analysis.** PBTO survives

Table 4. **Defense Stress Test (CIFAR-10).** We evaluate the ASR of PBTO (Task 1 injection) against various defenses applied after the final task ($N = 5$). PBTO maintains high efficacy across both mitigation and purification paradigms, demonstrating robust structural anchoring.

Defense Mechanism	ASR (%)
No Defense (Baseline)	97.1±0.4
Fine-Pruning (FP)	88.4±1.2
NAD (Distillation)	82.1±1.8
BTI-DBF (Decoupling)	89.5±0.9
REFINE (Reconstruction)	86.2±1.5

Pruning (88.4% ASR) because the trigger hijacks high-utility structural filters (e.g., edge detectors) that are critical for benign accuracy and thus cannot be pruned. It resists **Decoupling** defenses like BTI-DBF (89.5% ASR) because our feature entanglement objective forces the trigger to reside within the target class's texture manifold, creating a dilemma for the defense: accept the trigger or destroy image semantics.

4.8. Ablation Study

In this section, we analyze the individual components and hyperparameters of PBTO. **Contribution of Key Compo-**

nents. We dismantle PBTO into (1) *Surrogate Trajectory Simulation* (Sim) and (2) *Subspace Constraint* ($\mathcal{L}_s$) in Table 5. Without simulation (Baseline), ASR collapses (4.5%) due to catastrophic forgetting. Using *Sim* alone improves persistence (68.9%) by anticipating parameter drift. However, the *Subspace Constraint* is crucial to reach optimal performance (86.5%) by actively anchoring the trigger to structural invariants that remain stable across model updates.

Efficiency and Robustness Analysis. To validate PBTO's practicality, we conducted extensive sensitivity analyses (detailed numerical tables are provided in **Appendix D**). **Data Efficiency.** PBTO demonstrates remarkable independence from large-scale data. Our experiments show that using only **1,000 images per proxy class** yields a respectable ASR of 76.8%. Performance saturates at 5,000 images, confirming that PBTO captures generic structural patterns without requiring the victim's full training set. **Memory Buffer Sensitivity.** We challenged PBTO under severe memory constraints where catastrophic forgetting is exacerbated. Even when the buffer size is reduced to $\mathcal{M} = 500$, PBTO maintains an ASR of **10.4%**, significantly outperforming baselines (e.g., BadNets at 1.2%) which collapse entirely. At $\mathcal{M} = 1000$, PBTO recovers to 69.2%, proving its efficacy is not solely reliant on replay abundance. **Hyperparameters.** Attack performance saturates after approximately 50 rounds of iterative refinement. Furthermore, the subspace regularization weight $\lambda = 1.0$ is optimal; excessive constraints ($\lambda > 5$) reduce trigger distinctiveness.

Table 5. **Component Analysis.** We decouple the Surrogate Trajectory Simulation (Sim) and Subspace Constraint ($\mathcal{L}_s$). Both components are essential for optimal persistence.

Sim (Trajectory)	$\mathcal{L}_s$ (Subspace)	ASR (%)	ACC (%)
✗	✗	4.5	68.8
✓	✗	68.9	67.5
✗	✓	62.1	66.9
✓	✓	**86.5**	67.2

4.9. Mechanism: Feature Space Visualization

Figure 4 visualizes the penultimate layer features after learning all tasks. PBTO achieves deep **Semantic Entanglement**: the feature embeddings of poisoned samples perfectly overlap with the cluster of benign target samples. This confirms that the PBTO trigger mimics high-utility structural invariants. Since the CIL model must preserve these structural features to prevent catastrophic forgetting of the old task, it inevitably preserves the backdoor associated with them.

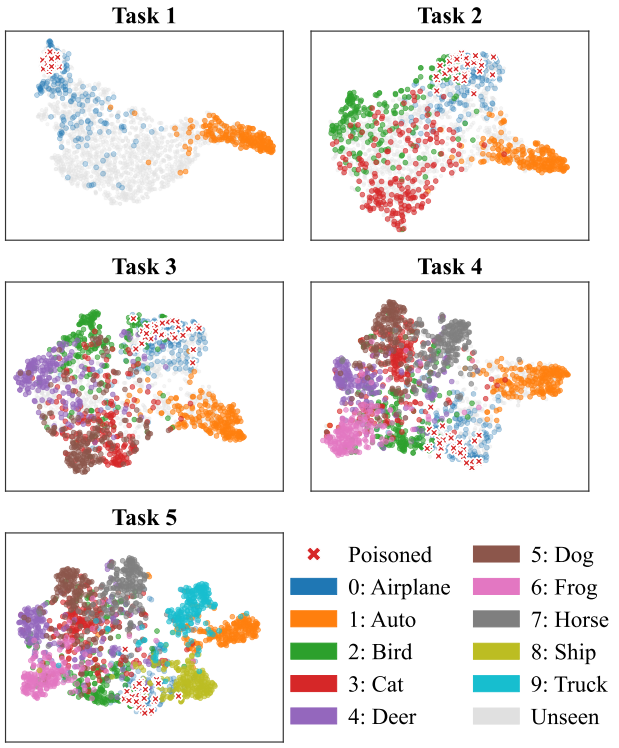


Figure 4. **Feature Space Visualization (t-SNE) after learning all N tasks.** PBTO poisoned samples (Red) are inextricably entangled with the benign target class manifold (Blue), confirming structural anchoring.

5. Conclusion

In this paper, we challenged the Local Stability Assumption prevalent in CL backdoor research. By identifying the limitations of single-task analysis in CIL, we proposed PBTO, a proactive framework utilizing trajectory simulation. By exposing the Fallacy of Local Stability, we demonstrate that the prevailing reactive paradigm—which relies on the availability of redundant capacity—is fundamentally brittle in class-incremental learning. Our results imply that true persistence in realistic, dynamic environments cannot be found by analyzing the current model state, but must be forged through future trajectory simulation. PBTO thus represents a necessary shift from finding stability to engineering it, establishing a new baseline for robustness where feature repurposing is the norm.

Impact Statement

This paper presents work whose goal is to advance the field of Machine Learning. There are many potential societal consequences of our work, none which we feel must be specifically highlighted here.

References

Buzzega, P., Boschini, M., Porrello, A., Abati, D., and Calderara, S. Dark experience for general continual learning: a strong, simple baseline. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 33, pp. 15920–15930, 2020.

Chen, L., Zhang, H., Wei, P., et al. Refine: Inversion-free backdoor defense via model reprogramming. In *International Conference on Learning Representations (ICLR)*, 2025.

Chen, X., Liu, C., Li, B., Lu, K., and Song, D. Targeted backdoor attacks on deep learning systems using data poisoning. In *arXiv preprint arXiv:1712.05526*, 2017.

Doan, K., Lao, Y., Zhao, W., and Li, P. LIRA: Learnable, imperceptible and robust backdoor attacks. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 34, pp. 11966–11978, 2021.

Gao, R. and Liu, W. Red alarm: Controllable backdoor attack in continual learning. *Neural Networks*, pp. 107479, 2025.

Gatys, L. A., Ecker, A. S., and Bethge, M. Image style transfer using convolutional neural networks. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 2414–2423, 2016.

Gu, T., Dolan-Gavitt, B., and Garg, S. Badnets: Identifying vulnerabilities in the machine learning model supply chain. In *Proceedings of the 2nd Workshop on Deep Learning and Security*, DLS '17, 2017.

Guo, Z., Kumar, A., and Tourani, R. Persistent backdoor attacks in continual learning. In *34th USENIX Security Symposium (USENIX Security 25)*, pp. 6379–6397, 2025.

Ho, J., Jain, A., and Abbeel, P. Denoising diffusion probabilistic models. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 33, pp. 6840–6851, 2020.

Hyvärinen, A. and Hoyer, P. Emergence of phase-and shift-invariant features by decomposition of natural images into independent feature subspaces. *Neural computation*, 12(7):1705–1720, 2000.

Jiang, W., Zhang, T., Qiu, H., Li, H., and Xu, G. Incremental learning, incremental backdoor threats. *IEEE Transactions on Dependable and Secure Computing*, 21 (2):559–572, 2022.

Kirkpatrick, J., Pascanu, R., Rabinowitz, N., et al. Overcoming catastrophic forgetting in neural networks. *Proceedings of the National Academy of Sciences*, 114(13): 3521–3526, 2017.

Kornblith, S., Norouzi, M., Lee, H., and Hinton, G. Similarity of neural network representations revisited. In *International conference on machine learning*, pp. 3519–3529. PMlR, 2019.

Li, Y., Lyu, X., Koren, N., Lyu, L., Li, B., and Ma, X. Neural attention distillation: Erasing backdoor triggers from deep neural networks. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 34, pp. 19631–19642, 2021.

Li, Y., Jiang, Y., Li, Z., and Xia, S.-T. Backdoor learning: A survey. *IEEE transactions on neural networks and learning systems*, 35(1):5–22, 2022.

Li, Z. and Hoiem, D. Learning without forgetting. *IEEE transactions on pattern analysis and machine intelligence*, 40(12):2935–2947, 2017.

Liu, K., Dolan-Gavitt, B., and Garg, S. Fine-pruning: Defending against backdooring attacks on deep neural networks. In *International Symposium on Research in Attacks, Intrusions, and Defenses (RAID)*, pp. 273–294. Springer, 2018.

Nguyen, T. A. and Tran, T. Wanet - imperceptible warping-based backdoor attack. In *International Conference on Learning Representations (ICLR)*, 2021.

Rebuffi, S.-A., Kolesnikov, A., Sperl, G., and Lampert, C. H. iCaRL: Incremental classifier and representation learning. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 2001–2010, 2017.

Wang, B., Yao, Y., Shan, S., Li, H., Viswanath, B., Zheng, H., and Zhao, B. Y. Neural cleanse: Identifying and mitigating backdoor attacks in neural networks. In *2019 IEEE symposium on security and privacy (SP)*, pp. 707–723. IEEE, 2019.

Wang, Z., Zhang, Z., Ebrahimi, S., Sun, R., Zhang, H., Lee, C.-Y., Ren, X., Su, G., Perot, V., Dy, J., et al. Dualprompt: Complementary prompting for rehearsal-free continual learning. In *European Conference on Computer Vision (ECCV)*, pp. 631–648. Springer, 2022a.

Wang, Z., Zhang, Z., Lee, C.-Y., et al. Learning to prompt for continual learning. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 139–149, 2022b.

Xu, Y., Wang, Z., Li, T., et al. Towards reliable and efficient backdoor trigger inversion via decoupling. In *International Conference on Learning Representations (ICLR)*, 2024.

Zeng, Y., Park, W., Mao, Z. M., and Jia, R. Rethinking the backdoor attacks' triggers: A frequency perspective. In

Proceedings of the IEEE/CVF international conference on computer vision, pp. 16473–16481, 2021.

A. Appendix

B. Detailed Experimental Setup

B.1. Hybrid Proxy Dataset Construction

As mentioned in Section 4.4, we employ a hybrid strategy to construct the proxy dataset $\tilde{\mathcal{D}}$ without accessing the victim's private data:

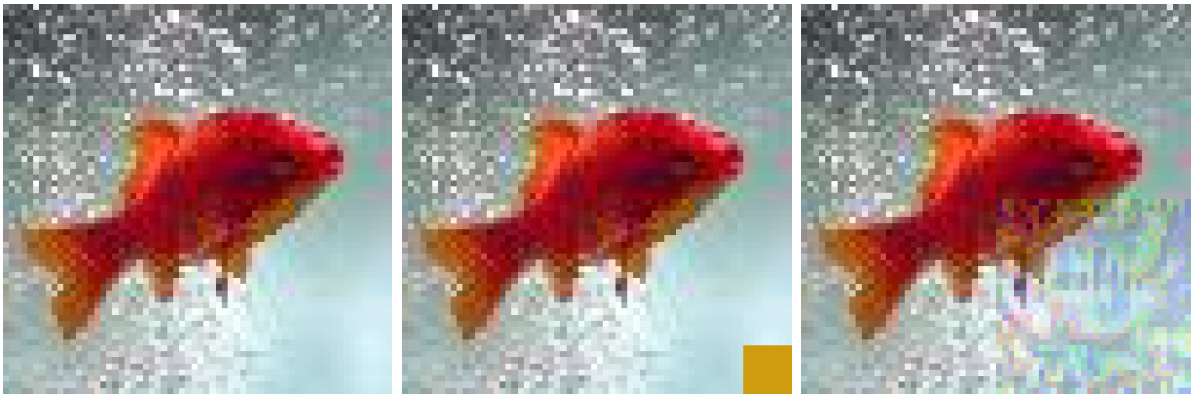
1. **Public Repositories:** We collect a subset of images from publicly available web-crawled datasets (e.g., unlabeled conceptual sets).

2. **Synthetic Generation (DDPM):** To enhance intra-class diversity, we utilize a pre-trained Denoising Diffusion Probabilistic Model (DDPM) (Ho et al., 2020) to generate synthetic samples.

For our standard experiments, we ensure a balanced distribution of 5,000 images per proxy class.

C. Visual Stealthiness Comparison

We provide a qualitative evaluation of trigger stealthiness in Figure 5. Unlike traditional attacks such as BadNets which introduce conspicuous visible patches, the poisoned samples generated by PBTO are visually indistinguishable from benign images. This demonstrates that PBTO exhibits a low probability of detection by human inspection.



(a) Clean Image (b) BadNets (c) PBTO

Figure 5. **Visual Stealthiness Comparison.** The poisoned sample generated by PBTO (c) is visually imperceptible relative to the benign image (a).

D. Extended Ablation Studies

We provide the detailed numerical results for the sensitivity analyses discussed in Section 4.8.

D.1. Data Efficiency and Buffer Size Tables

Table 6 and Table 7 provide the quantitative results for data efficiency and memory buffer sensitivity, respectively.

Table 6. **Impact of Proxy Data Size (Per Class).** PBTO maintains high efficacy even with a moderate number of images.

Size	1,000	2,000	**5,000**	10,000
ASR (%)	76.8	82.4	**86.5**	87.1

Table 7. **Sensitivity to Memory Buffer Size ($\mathcal{M}$).** (CIFAR-100, Task 10). PBTO outperforms baselines under strict constraints.

Method	Buffer Size ($\mathcal{M}$)			
	500	1,000	2,000	5,000
BadNets	1.2%	2.5%	4.5%	12.8%
WaNet	3.8%	4.4%	3.2%	10.5%
LTB	3.6%	3.8%	4.7%	6.2%
Ours	**10.4%**	**69.2%**	**86.5%**	**91.3%**

D.2. Simulation Trajectory Length (M)

Table 8 confirms that a short trajectory ($M = 1$) causes overfitting to the immediate snapshot, while $M = 5$ is sufficient to capture robust invariants.

Table 8. **Impact of Simulation Trajectory Length (M).**

Trajectory (M)	1	3	5	10
ASR (%)	65.2	82.4	**86.5**	86.5

D.3. Impact of Iterative Refinement and λ

Figure 6 (Left) shows the ASR improvement over refinement rounds. The attack rises rapidly and saturates after approximately 50 rounds. Figure 6 (Right) shows the sensitivity to λ, confirming an inverted U-shape curve peaking at 1.0.

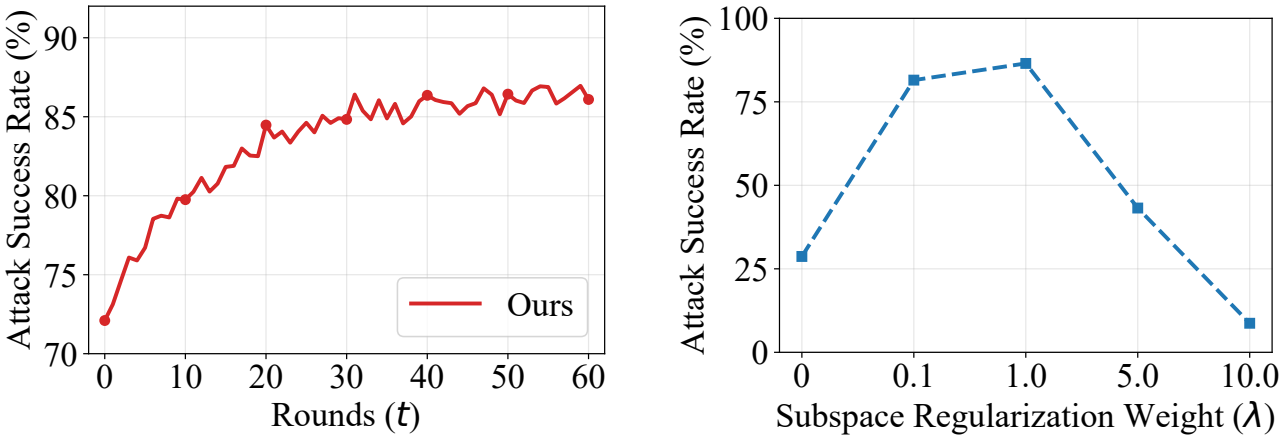


Figure 6. **Left:** Impact of Iterations. **Right:** Sensitivity to λ.