INFLUENCE-GUIDED ACTIVE SEARCH FOR POISONED TRAINING DATA FORENSICS

Anonymous authors
Paper under double-blind review

ABSTRACT

Data poisoning attacks that inject malicious samples into training data pose a serious threat to the reliability of machine learning. Existing defenses focus on automated detection and removal, but effective cleaning inevitably discards significant amounts of benign data. We instead consider *forensic investigations of poisoned data*, which verify samples through manual inspection or provenance checks, such as re-acquiring a sample from its original source or auditing its origin. A key challenge of forensic investigations is that verification is expensive and can be performed only per sample, but investigation budgets are limited. Therefore, an investigation must strategically select the samples that will be verified and potentially removed to minimize the impact of the remaining poisons. We frame this as a nonmyopic sequential search problem and introduce an *influence-guided active search* (IGAS) framework that integrates: (i) a label-free influence score identifying samples with disproportionate impact on predictions and (ii) an adaptive search strategy propagating information from verified samples to focus on regions of the data that are both influential and likely to be poisoned. Experiments on CIFAR-10, Tiny ImageNet, and GTSRB across eight triggerless and backdoor attacks demonstrate that IGAS substantially reduces Attack Success Rate while preserving test accuracy, outperforming automated data-cleaning defenses that often remove benign samples along with poisoned ones.

1 INTRODUCTION

Data poisoning attacks pose a significant threat to the reliability and security of machine learning systems. By injecting malicious samples into the training data, an adversary can influence the behavior of the trained model, causing targeted misclassification or degrading model performance. This risk is especially relevant in modern learning pipelines that rely on data collected from web sources, third-party providers, or crowdsourcing platforms, where the full provenance of every training sample may be difficult to verify (Gu et al., 2019; Goldblum et al., 2022). Prior work has shown that even a small fraction of poisoned samples can induce targeted failures while leaving overall test accuracy largely unchanged (Carlini & Terzis, 2022).

Clean-label targeted poisoning attacks are particularly challenging because poisoned samples retain plausible labels and often appear visually similar to benign examples. Triggerless attacks, instead of changing the label, modify the input so that poisoned samples influence the learned representation or decision boundary in favor of an attacker-chosen target (Fan et al., 2022; Aghakhani et al., 2021; Geiping et al., 2021). Backdoor attacks introduce a related threat by training the model to associate a hidden trigger pattern with a malicious prediction, while preserving normal behavior on clean inputs (Zeng et al., 2023b; Huynh et al., 2024; Nguyen et al., 2025). Because these attacks may not produce obvious input-level or loss-based anomalies, they can be difficult to detect using standard data filtering or label-checking methods (Turner et al., 2018; Tran et al., 2018; Jagielski et al., 2018).

Existing defenses have made significant progress in detecting or mitigating poisoned data, including data sanitization, robust training, and influence-based filtering methods (Chen et al., 2021; Levine & Feizi, 2021; Van et al., 2023). However, fully automated defenses often face a fundamental trade-off. Conservative filtering may leave influential poisons in the training set, while aggressive filtering can remove many benign samples and reduce clean accuracy. This trade-off is especially problematic

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

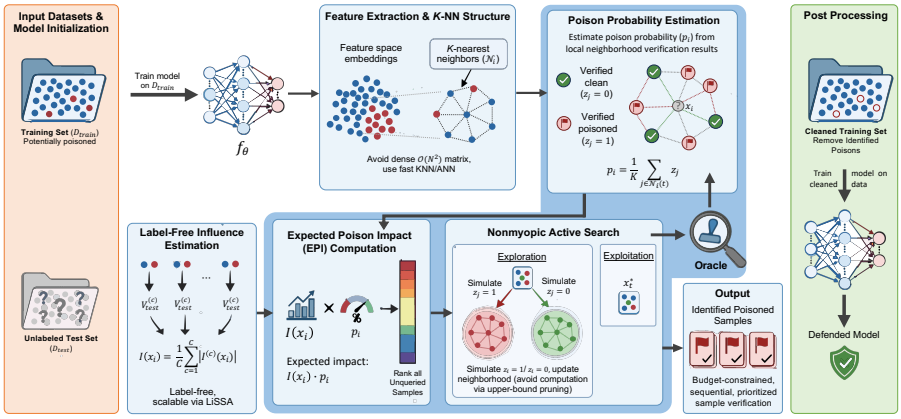


Figure 1: Overview of IGAS. Given a potentially poisoned training set, unlabeled test inputs, and a limited verification budget, IGAS combines poison probability with label-free influence to prioritize samples for verification. Verified poisons are removed before final training to reduce attack success while preserving benign data.

when poisoned samples are designed to resemble benign data or when they lie near informative decision-boundary regions (Koh et al., 2022).

In this paper, we study a complementary security setting: *forensic investigations of poisoned training data*. Instead of assuming that suspicious samples can be automatically discarded, we consider defenders who must justify sample-level removal decisions using an expensive and potentially imperfect verification process. Verification does not necessarily rely on *manual inspection* and detection of perturbations; it may also check the *provenance* of a sample, for example, by requesting the original from the data provider and comparing the two, or by auditing its source and collection record. Such verification actions are typically available only one sample at a time and at non-trivial cost (rate-limited APIs, provider responses, expired links), so exhaustively verifying a large training set is impractical (Carlini et al., 2024). This forensic setting arises in security-sensitive ML pipelines, where incorrect removal of benign data can degrade the accuracy and usefulness of the model, but retaining even a small number of influential poisons may lead to a poisoned, compromised model. The defender's goal is therefore not only to detect poisoned samples, but to prioritize verification toward samples whose removal most improves model integrity under a limited investigation budget.

This forensic setting introduces three challenges. The selection policy must be (i) *sample-efficient*, since only a small fraction of the dataset can be verified; (ii) *computationally tractable* for deep models, where repeated retraining or exact second-order influence computation is expensive; and (iii) *impact-aware*, since not all poisoned samples are equally harmful, and removing a small number of highly influential poisons may be more useful than identifying many low-impact ones.

Active search methods aim to find members of a rare target class, termed "positive instances," under a limited query budget (Garnett et al., 2012; Jiang et al., 2017). For example, in drug discovery, positive instances are biologically active compounds sought through sequential screening (Garnett et al., 2015). These studies focus on discovery tasks rather than identifying poisoned training samples. Their standard objective is to maximize the number of positive instances discovered, treating each positive instance as equally valuable. This objective is insufficient for poisoning defense, where poisons differ in their effects on model behavior, and a small number of highly influential poisons may be critical to attack success. Consequently, a defender should prioritize samples based both on their probability of being poisoned and the expected benefit of identifying and removing them.

Our key observation is that harmful poisoned samples often exert asymmetric, class-specific influence on test-time decision boundaries, even when their labels and input appearance remain plausible. On CIFAR-10, poisoned samples have 678–1,199$\times$ higher mean influence than clean samples, and they rank, on average, within the top 6% of the training set (Appendix D); moreover, a class-wise influence signal tracks test influence far more closely than aggregate label-free alternatives (Spearman correlation 0.62–0.87 vs. at most 0.42; Appendix E.5). This suggests that poisoned-sample discovery should not rely only on generic anomaly scores or aggregate uncertainty measures. Instead, the

defender should estimate how strongly each training sample can affect the model's behavior on the test inputs, while remaining agnostic to the attacker's target class.

To address the three challenges of the forensic setting, we propose **Influence-Guided Active Search (IGAS)**, a forensic investigation framework for prioritizing the investigation of training samples under a limited verification budget (Figure 1). Our contributions are threefold. First, we formulate poisoned-data defense as a forensic investigation problem for security-sensitive ML pipelines, where reliable sample-level verification is available but expensive (Section 3). Second, we introduce a label-free, class-wise influence proxy that estimates the security impact of training samples using only unlabeled test inputs (Section 4.2). We construct class-wise impact vectors from unlabeled test data, avoiding the labeled validation-data requirement of classical influence methods (Koh & Liang, 2017; Basu et al., 2021). Third, we design IGAS, a nonmyopic influence-guided active search framework that prioritizes samples according to both their probability of being poisoned and their influence, so that verification prioritizes samples that have high expected utility in terms of their potential for malicious impact (Sections 4.3–4.4). IGAS adapts nonmyopic active search to high-dimensional visual representations through neighborhood-based probability updates and optimistic pruning, enabling the policy to reason about future queries while maintaining computational scalability.

We evaluate IGAS on CIFAR-10 (Krizhevsky et al., 2009), Tiny ImageNet (Le & Yang, 2015), and GTSRB (Stallkamp et al., 2011) against both triggerless and backdoor attacks, including Feature Collision, Bullseye Polytope, Gradient Matching, Narcissus, COMBAT, and Wicked variants (Sections 5). IGAS reduces attack success rates under a small verification budget while preserving clean accuracy. With a budget of only 0.5% of the CIFAR-10 training set, IGAS reduces ASR to at most 5% across all eight clean-label settings (from up to 100% without defense), while achieving the highest clean accuracy among defenses in every setting; at the same budget, 86–94% of its queries are true poisons, compared with at most 8% of the samples removed by automated baselines (Appendix E.6). We discuss limitations in Section 6.

2 Related Work

2.1 Data Poisoning Attacks

Data poisoning attacks compromise machine learning systems by injecting malicious samples into the training dataset (Fan et al., 2022; Wang et al., 2024; Biggio et al., 2013). Even a small number of poisoned samples can induce targeted misclassification or degrade model performance while preserving test accuracy (Carlini & Terzis, 2022; Guo & Liu, 2020).

There are two main types of clean-label targeted attacks. First, triggerless attacks keep labels unchanged while modifying inputs, so that poisoned samples influence the learned representation or decision boundary (Shafahi et al., 2018; Aghakhani et al., 2021; Geiping et al., 2021). Since labels remain plausible and inputs often appear benign, these attacks are difficult to detect through label verification or simple anomaly detection (Peri et al., 2020; Chen et al., 2021; Goldblum et al., 2022; Tian et al., 2022; Ramirez et al., 2022). Next, backdoor attacks associate a trigger pattern with an attacker-chosen prediction while preserving normal behavior on clean inputs (Zeng et al., 2023b; Gu et al., 2019; Chen et al., 2017; Liu et al., 2018; Turner et al., 2019). Unlike label-flipping attacks (Xiao et al., 2012), both types of these attacks can preserve label correctness, making them harder to identify through standard data inspection.

2.2 Data Poisoning Defenses

Existing defenses include data sanitization, robust training, and influence-based filtering. Data sanitization methods attempt to identify suspicious samples before or during training (Steinhardt et al., 2017; Koh et al., 2022), using spectral or representation-based anomaly detection (Tran et al., 2018; Chen et al., 2018; Yang et al., 2022) or local-neighborhood consistency checks such as Deep k-NN (Peri et al., 2020). Similarly, Meta-Sift (Zeng et al., 2023a) sifts out a small, high-precision clean subset from a poisoned dataset, which can then support downstream defenses that require trusted data. Influence-based defenses estimate how individual training samples affect downstream model behavior and remove samples with high estimated harm (Seetharaman et al., 2022; Steinhardt et al., 2017). Robust training methods, such as HINT (Van et al., 2023), instead aim to reduce the effect of harmful samples without explicitly removing them.

These defenses often assume that suspicious samples can be removed automatically, creating a practical trade-off. Conservative filtering may leave influential poisons in the training set, while aggressive filtering may remove many benign samples and reduce clean utility. Influence-based methods often require labeled validation data or knowledge of the target behavior, which may be unavailable in deployment. Post-hoc forensic methods provide another direction by tracing observed model failures back to responsible training samples (Shan et al., 2022), but they typically assume that a misclassified input or suspicious behavior has already been observed. In contrast, we study a pre-training forensic setting in which a reliable but expensive oracle can verify only a small number of samples, so the key question is which samples to verify rather than which to remove. Unlike classical active search (Jiang et al., 2017), which treats all discovered positive instances equally, IGAS weights each discovery by its estimated influence on model behavior.

3 Problem Setting and Threat Model

We study a pre-training forensic investigation setting: a defender has collected a training set that may contain malicious samples and must decide which samples to verify before training the deployed model. The defender trains a parametric model f_θ on $D_{\text{train}} = \{(x_i, y_i)\}_{i=1}^N$ and has access to an unlabeled test set $D_{\text{test}} = \{x_j\}$ of inputs on which the model should behave reliably. Appendix A summarizes notation.

3.1 Threat Model

Following standard poisoning settings (Shafahi et al., 2018; Geiping et al., 2021; Zeng et al., 2023b), we specify the attacker's goal, capability, and knowledge.

Attacker's Goal. The attacker aims to compromise the integrity of a model trained on D_{train}. In triggerless attacks, the attacker perturbs selected training inputs while preserving labels so that attacker-chosen test inputs are misclassified. In backdoor attacks, the attacker injects triggered training samples so that the model predicts an attacker-chosen class when the trigger appears at inference time, while behaving normally on clean inputs.

Attacker's Capability. The attacker can modify or inject up to δN training samples, where δ denotes the poisoning rate, by applying a bounded perturbation of size ϵ to each input while keeping its label unchanged. The defender does not know which samples are poisoned before verification.

Attacker's Knowledge. Following common evaluation settings in the poisoning defense literature, the attacker knows the model architecture and training procedure, which allows evaluation under relatively strong adversarial conditions.

3.2 Defender Setting and Verification Problem

The defender has access to the potentially contaminated training set D_{train} and unlabeled test set D_{test}. The defender also has an expensive verification process for individual training samples and must select which samples to verify given a limited budget $B \ll N$. Our formulation assumes correct verification outcomes; Appendix E.7 shows that IGAS remains effective with imperfect oracles.

Constraint 1: No Clean Labeled Validation Set. Unlike defense settings that assume access to trusted clean data for validation (Qi et al., 2023; Wang et al., 2019; Zeng et al., 2021), the defender has neither a labeled clean validation set nor a trusted clean subset of training data. Only the unlabeled test inputs in $D_{\text{test}} = \{x_j\}$ are available. This makes classical influence estimation, which requires gradients of a labeled evaluation loss, impossible to apply directly.

Constraint 2: Limited Verification Budget. The defender can verify only a small subset of training samples. Verification determines whether a sample violates integrity or provenance constraints by comparing it against a trusted reference rather than by judging its appearance. Examples include re-acquiring the sample from its original source and checking whether it was modified, requesting the original from the data provider, and auditing the sample's origin (e.g., uploader, domain, or collection log). Because the check compares with a reference, it applies even when the perturbation is imperceptible. The reference is, however, accessible only per sample and at non-trivial cost, which is why exhaustive verification is infeasible and the budget B binds. Each query returns a binary status

$z_i \in \{0, 1\}$, where $z_i = 1$ indicates that sample x_i is confirmed to be poisoned and $z_i = 0$ that it is verified to be clean. The defender thus aims to identify as many influential poisons as possible within the budget. This setting complements automated defenses. Automated filtering can screen large datasets but may remove benign data or miss influential poisons depending on its threshold. Selective verification suits pipelines that require higher confidence sample-level decisions, e.g., those involving external data providers, provenance records, or high-stakes deployment settings.

3.3 Objective

The defender sequentially verifies one sample at a time, updating its policy after each outcome, so that removing verified poisons reduces the attack's effect while preserving benign data. Ideally, it would select a set S whose verified poisons, once removed, yield the best model behavior:

$$S^* = \underset{S \subseteq D_{\text{train}}, |S| \leq B}{\arg\min} \mathcal{L}\left(f_\theta(D_{\text{train}} \setminus \{x_i \in S \mid z_i = 1\}), D_{\text{test}}\right). \tag{1}$$

This objective cannot be directly optimized for multiple reasons. First, the test loss is unavailable without labels for D_{test}. In addition, the defender also does not know which queried samples have $z_i = 1$ until verification. We therefore use an influence-based surrogate that, if the poison indicators were known, would prioritize verifying poisons based on their estimated impact on model behavior:

$$S^* = \underset{S \subseteq D_{\text{train}}, |S| \leq B}{\arg\max} \sum_{x_i \in S} z_i \cdot \mathcal{I}(x_i), \tag{2}$$

where $\mathcal{I}(x_i)$ denotes the estimated influence of sample x_i on model predictions. Since z_i is unknown before verification, a natural approximation replaces it with an estimated poison probability, yielding an expected poison impact score that favors samples both likely to be poisoned and influential.

Even so, exact optimization is infeasible, and a practical policy faces three challenges. First, the number of possible oracle outcome sequences grows exponentially with the budget, since each queried sample may be verified as poisoned or clean. Second, evaluating each sequence would require retraining or re-evaluating the model on the resulting filtered dataset, which is prohibitively expensive at training-set scale. Third, the query policy should be nonmyopic: greedily selecting the sample with the highest expected poison impact ignores how its verification outcome may change the verified set and the subsequent probability estimates. These challenges motivate IGAS, an approximate nonmyopic policy that combines label-free influence estimation, poison-probability updates, and lookahead-based non-myopic active search.

4 Influence-Guided Active Search

IGAS operationalizes the surrogate in Eq. (2) by replacing the unknown poison indicator z_i with an estimated poison probability p_i, yielding the *Expected Poison Impact* (EPI), $p_i \cdot \mathcal{I}(x_i)$. Since optimizing EPI over all query sequences remains intractable, IGAS uses two approximations: a label-free influence estimate that requires neither labeled test data nor exact Hessian inversion (Section 4.2), and neighborhood-based probability propagation with optimistic upper-bound pruning that makes nonmyopic search tractable in high-dimensional feature spaces (Sections 4.3–4.4). Both components operate on a shared feature space (Section 4.1).

4.1 Construction of Feature Space and Distance Matrix

We first train the base model f_θ on the potentially poisoned training set D_{train}, which then serves both as the reference model for influence estimation (Section 4.2) and as a feature extractor: each sample $x_i \in D_{\text{train}}$ is represented by its penultimate-layer embedding. At each search step t, let $\mathcal{N}_i(t)$ denote the K nearest verified samples to x_i within the queried set S_t, measured by Euclidean distance in this feature space. The set of neighbors changes as more samples are verified and provide localized evidence for poison-probability estimation. The pairwise distance matrix is precomputed once after feature extraction, enabling efficient exact neighbor retrieval for moderate-scale datasets as S_t grows. For larger datasets where storing all pairwise distances becomes prohibitive, the same procedure can use approximate nearest-neighbor methods such as FAISS (Douze et al., 2026) or HNSW (Malkov & Yashunin, 2018), avoiding explicit storage of the complete distance matrix while maintaining scalable neighborhood updates.

4.2 LABEL-FREE INFLUENCE FORMULATION

Classical influence functions (Koh & Liang, 2017) estimate how individual training samples affect model behavior, but they require labeled test or validation data to compute the test-loss gradient, which is unavailable in our setting. Therefore, we introduce a target-agnostic and label-free influence formulation that replaces this gradient with class-wise test impact vectors, $V_{\text{test}}^{(c)}$. For each class $c \in \{1, \ldots, C\}$, we assign every unlabeled test instance to the pseudo-label c, and we compute the loss gradient:

$$V_{\text{test}}^{(c)} = \nabla_\theta \left(\frac{1}{|D_{\text{test}}|} \sum_{x_j \in D_{\text{test}}} \mathcal{L}\left(f_\theta(x_j), c\right) \right), \tag{3}$$

where $f_\theta(x_j)$ represents the network output. $V_{\text{test}}^{(c)}$ captures the sensitivity of the class-c pseudo-loss over D_{test} to changes in model parameters. Unlike aggregate label-free metrics, such as the L_1 or L_2 norms of the output logits or the entropy of the predicted softmax distribution, which summarize model behavior as a single scalar, this formulation estimates how parameter perturbations affect the model output with respect to each candidate class. Thus, it better captures the asymmetric, class-specific effects induced by targeted poisoning and backdoor attacks (Table 12 in Appendix E.5).

Leveraging the foundational first-order Taylor approximations of the influence framework, we project the parameter perturbation $\Delta\theta_i$ induced by removing x_i onto $V_{\text{test}}^{(c)}$:

$$\mathcal{I}^{(c)}(x_i) = (\Delta\theta_i)^\top V_{\text{test}}^{(c)} \approx -\left(\nabla_\theta \mathcal{L}(x_i, \theta)\right)^\top H_\theta^{-1} V_{\text{test}}^{(c)}. \tag{4}$$

Because the defender does not know the attacker's target class, the direction of influence is unknown. So, we treat both positive and negative changes as indicators of substantial effect and aggregate the class-wise values into a target-agnostic score using the mean absolute influence:

$$\mathcal{I}(x_i) = \tfrac{1}{C} \textstyle\sum_{c=1}^{C} \left|\mathcal{I}^{(c)}(x_i)\right|. \tag{5}$$

A large $\mathcal{I}(x_i)$ indicates that x_i strongly influences the model's predictions on unlabeled test data, rather than its loss on labeled test data as in classical influence.

Since computing the exact inverse Hessian H_θ^{-1} costs $\mathcal{O}(P^3)$, we approximate the inverse Hessian-vector product $H_\theta^{-1} V_{\text{test}}^{(c)}$ using LiSSA, which applies a truncated Neumann series expansion (Van et al., 2023; Agarwal et al., 2017). These C vectors are computed once before the investigation and cached, so scoring any training sample reduces to a dot product with its gradient $\nabla_\theta \mathcal{L}(x_i, \theta)$. The cost thus remains linear in the training set size, with only an additional linear dependence on C.

4.3 POISON PROBABILITY ESTIMATION

At each step t, the dataset is divided into the queried set S_t, which has been verified by the oracle, and the unqueried set $U_t = D_{\text{train}} \setminus S_t$. For each $x_i \in U_t$, we estimate its poison probability as the fraction of its neighborhood $\mathcal{N}_i(t)$ that is verified to be poisoned:

$$p_i = \tfrac{1}{K} \textstyle\sum_{j \in \mathcal{N}_i(t)} z_j, \tag{6}$$

where $z_j \in \{0, 1\}$ is the poison status returned by the oracle, and K is the neighborhood size ($K = 10$ is chosen empirically in Appendix E.2). This propagates oracle feedback from verified samples to their unverified neighbors. Each unqueried sample is then prioritized by its EPI score $p_i \cdot \mathcal{I}(x_i)$, which favors samples that are both likely to be poisoned and influential on model behavior.

Initial Seeding Strategy At $t = 0$, $S_0 = \emptyset$ and all $p_i = 0$. To bootstrap the search, IGAS first queries the K samples with the highest influence scores $\mathcal{I}(x_i)$, providing verified samples from which subsequent probability estimates are computed. This matters when the poisoning rate δ is small. For $\delta = 1\%$ and $K = 10$, a random seed batch would contain no poisons with probability $(1 - 0.01)^{10} \approx 90.4\%$, leaving the neighborhood estimates without positive feedback. If all seeds are verified as clean, IGAS continues to rely on influence-guided ranking until poison evidence becomes available. In our experiments, this strategy often identifies a poisoned sample early in the search.

4.4 Sequential Search Algorithms

IGAS is a nonmyopic sequential policy for selecting which samples to verify. For ease of exposition, we first describe a myopic variant based only on the current EPI, then the nonmyopic policy that also accounts for how the current query may improve future decisions.

Myopic Search At each step t, the myopic variant selects the unqueried sample with the highest immediate EPI:

$$x_t^* = \arg\max_{x_i \in U_t} p_i \cdot \mathcal{I}(x_i). \tag{7}$$

This requires only a single pass over the current EPI scores, but ignores how discovering a poison may change the estimated probabilities of other samples in future steps.

Nonmyopic Influence-Guided Active Search To account for future query value, IGAS adapts Efficient Nonmyopic Search (ENS) (Jiang et al., 2017) to our influence-guided poisoning investigation setting. For each candidate sample $x_i \in U_t$, we compute a lookahead score T_i that combines the immediate expected poison impact of querying x_i with the expected benefit of future queries:

$$T_i = p_i \cdot \mathcal{I}(x_i) + p_i \cdot R_{\text{poison}} + (1 - p_i) \cdot R_{\text{clean}}. \tag{8}$$

where R_{poison} and R_{clean} are the estimated future EPI after a poisoned ($z_i = 1$) or clean ($z_i = 0$) outcome. To compute them, we simulate each outcome. A simulated poisoned outcome increases the poison probabilities of unqueried samples whose nearest verified neighborhoods include x_i, and a clean outcome decreases them. Let $p_k^{(i,z)}$ denote the updated probability of x_k after simulating $z_i = z$, and let $\text{TopSum}_{B-t-1}\{\cdot\}$ denote the sum of the $B - t - 1$ largest values in a set. Following ENS, we approximate the future reward by the highest-scoring remaining samples:

$$R_{\text{clean}} = \text{TopSum}_{B-t-1}\left\{ p_k^{(i,0)} \cdot \mathcal{I}(x_k) \,\middle|\, x_k \in U_t \setminus \{x_i\} \right\},$$
$$R_{\text{poison}} = \text{TopSum}_{B-t-1}\left\{ p_k^{(i,1)} \cdot \mathcal{I}(x_k) \,\middle|\, x_k \in U_t \setminus \{x_i\} \right\}.$$

Computing Equation (8) for every candidate in U_t at every iteration can be computationally expensive. To reduce this cost, we employ a pruning strategy based on optimistic upper bounds. Specifically, before simulating the two possible outcomes for any candidate, we calculate global upper bounds on future rewards from investigating the unqueried pool U_t. These bounds are also based on the sum of the $B - t - 1$ largest values, but over the entire unqueried pool U_t. For the poisoned case, we consider the maximum possible increase in probability p_k due to a newly verified poisoned neighbor, which is $1/K$. The two upper bounds are:

$$UB_{\text{clean}} = \text{TopSum}_{B-t-1}\left\{ p_k \cdot \mathcal{I}(x_k) \,\middle|\, x_k \in U_t \right\},$$
$$UB_{\text{poison}} = \text{TopSum}_{B-t-1}\left\{ \left(p_k + \frac{1}{K} \right) \cdot \mathcal{I}(x_k) \,\middle|\, x_k \in U_t \right\}.$$

Unlike the lookahead terms R_{clean} and R_{poison}, these bounds are computed over the entire unqueried pool U_t and therefore do not exclude the candidate x_i. For each candidate x_i, we compute an optimistic score $UB(i) = p_i \cdot \mathcal{I}(x_i) + p_i \cdot UB_{\text{poison}} + (1 - p_i) \cdot UB_{\text{clean}}$. If this score is lower than the best score found so far, T^*, we skip the full simulation for x_i. This pruning step reduces unnecessary lookahead computations while preserving the search over candidates that can still improve the current best score. Algorithm 1 in Appendix C provides detailed pseudocode for the IGAS approach.

5 Experiments

5.1 Datasets and Attack Setup

We evaluate on **CIFAR-10** (Krizhevsky et al., 2009) (50,000 training images, 10 classes) and **GTSRB** (Stallkamp et al., 2011) (39,270 traffic-sign images, 43 classes), using ResNet-18 (He et al., 2016) unless otherwise stated. Results on **Tiny ImageNet** (Le & Yang, 2015) with VGG-16 are reported in Appendix E.1.

We consider two types of clean-label attacks: triggerless and backdoor attacks. The triggerless attacks are **FC** (Shafahi et al., 2018) and **BP** (Aghakhani et al., 2021), which manipulate feature-space

geometry, and **GM** (Geiping et al., 2021), which matches poisoned-batch gradients to target gradients; we poison 1% of CIFAR-10 (500 images) with $\epsilon \in \{8, 16, 22\}$ and average over 20 runs with distinct base-target pairs for these attacks. The backdoor attacks are **Narcissus** (Zeng et al., 2023b), **COMBAT** (Huynh et al., 2024), and **Wicked** (Nguyen et al., 2025) applied to BadNets, Blended, and SIG triggers; we poison 0.5% of CIFAR-10 and GTSRB (accounting for class-size variations in GTSRB) with $\epsilon = 16$ and average over 10 runs with distinct target classes for these attacks.

FC and BP use transfer learning with a pre-trained ResNet-18 (40 epochs, learning rate 0.1 decayed at epochs 25 and 35); GM trains from scratch for 200 epochs (learning rate 0.1 decayed at epochs 100 and 150). Otherwise, poison generation and training follow the original configuration of each attack (Schwarzschild et al., 2021; Zeng et al., 2023b; Nguyen et al., 2025). All experiments run on NVIDIA RTX A5000 GPUs (24GB). We provide the source code as supplementary material.

5.2 Evaluation Metrics

After removing the samples identified as poisoned, we train a model on the cleaned dataset following the standard training protocol assumed by each attack. We then evaluate defense results using two metrics: Attack Success Rate (ASR) and clean test accuracy (Acc.). First, **Attack Success Rate (ASR)** measures the percentage of targeted test inputs that a poisoned model misclassifies into the attacker's chosen class. Second, **clean test accuracy (Acc.)** measures the classification accuracy of the model on the original clean test set. An effective defense should minimize ASR while keeping clean test accuracy high.

5.3 Baselines

We compare IGAS with six automated data-cleaning defenses, which remove suspicious samples without sample-level verification. For triggerless attacks, we use **Meta-Sift** (Zeng et al., 2023a) and **EPIC** (Yang et al., 2022). Meta-Sift is designed to identify a small clean subset rather than to remove poisons; it assigns each training sample a cleanliness score via splitting-based training with a meta-learned weighting network. EPIC removes samples isolated in gradient space during training. For backdoor attacks, we use **Activation Clustering (AC)** (Chen et al., 2018), which removes minority clusters of per-class activations; **SPECTRE** (Hayase et al., 2021), which detects anomalous spectral signatures via robust covariance estimation; and **FLARE** (Hou et al., 2025), which combines hidden-layer features to detect diverse trigger types. **Confusion Training** (Qi et al., 2023), which trains with randomly mislabeled clean samples so that only backdoor samples are fitted, is evaluated in both settings. Appendix E.6 provides an alternative comparison, which uses these baselines to select samples for verification and potential removal, under the same budget as IGAS.

5.4 Results

Triggerless Attacks. With a budget of $B = 250$ (0.5% of the CIFAR-10 training set), IGAS reduces ASR to at most 5% across all attacks and perturbation sizes (Table 1), down from 95–100% for BP, 75–100% for GM, and 15–35% for FC without defense. IGAS also achieves the highest clean accuracy among all defenses in every setting.

Backdoor Attacks. Table 2 shows that IGAS offers the best trade-off between ASR and clean accuracy. FLARE attains low ASR on most attacks, but removes substantial amount of benign data, reducing clean accuracy to 45.9–58.3% on CIFAR-10 and 84.0–89.4% on the GTSRB Wicked variants; we therefore exclude these entries from ranking. On CIFAR-10, IGAS retains at least 90.7% clean accuracy and achieves the lowest ASR on four of five attacks; it reduces Wicked-SIG ASR from 93.72% to 18.69%, while AC, SPECTRE, and Confusion Training remain above 76%. On GTSRB, IGAS achieves the lowest or second-lowest ASR on four of five attacks and the highest clean accuracy on all three Wicked variants; Wicked-SIG remains the most challenging (24.30%).

Additional Results. The appendix reports results on Tiny ImageNet (Appendix E.1); a budget-matched comparison with the baselines as forensic algorithms, in which 86–94% of IGAS selections are true poisons vs. at most 8% of the samples selected by Meta-Sift and EPIC (Appendix E.6); ablations of influence guidance and nonmyopic lookahead (Appendix E.4); sensitivity to hyperparameter K and the verification budget (Appendices E.2 and E.3); comparisons with true-label influence and aggregate label-free proxies (Appendix E.5); and robustness to noisy oracles (Appendix E.7).

Table 1: Post-defense attack success rate (ASR) and clean test accuracy (Acc.) under triggerless attacks on CIFAR-10. Results are averaged over 20 poisoned datasets across attack methods and perturbation sizes (ϵ). The lowest ASR and highest Acc. among defenses are highlighted in **bold**.

Attack		Undefended		Defended							
				IGAS		Meta-Sift		EPIC		Confusion Training	
Size	**Method**	ASR (↓)	Acc. (↑)	ASR (↓)	Acc. (↑)	ASR (↓)	Acc. (↑)	ASR (↓)	Acc. (↑)	ASR (↓)	Acc. (↑)
ϵ **= 8**	FC	15%	0.9460	**0%**	**0.9493**	10%	0.9423	**0%**	0.9467	5%	0.9474
	BP	100%	0.9457	**0%**	**0.9474**	20%	0.9455	5%	0.9467	45%	0.9416
	GM	75%	0.9487	**5%**	**0.9496**	20%	0.9407	**5%**	0.8899	10%	0.9040
ϵ **= 16**	FC	25%	0.9456	**0%**	**0.9476**	15%	0.9325	**0%**	0.9466	5%	0.9451
	BP	100%	0.9458	**0%**	**0.9480**	35%	0.9342	5%	0.9465	70%	0.9334
	GM	95%	0.9486	**5%**	**0.9475**	40%	0.9352	30%	0.8900	15%	0.9051
ϵ **= 22**	FC	35%	0.9455	**0%**	**0.9468**	15%	0.9456	**0%**	0.9466	20%	0.9417
	BP	95%	0.9458	**5%**	**0.9504**	55%	0.9386	**5%**	0.9468	25%	0.9500
	GM	100%	0.9491	**5%**	**0.9490**	20%	0.9373	20%	0.8901	15%	0.9122

Table 2: Post-defense attack success rate (ASR) and clean test accuracy (Acc.) under backdoor attacks on CIFAR-10 and GTSRB, averaged over 10 runs. Best and second-best results among defenses are highlighted in **bold** and underlined, respectively. Entries marked with † are excluded from the ranking. Note that FLARE removes a large portion of benign training data, reducing clean accuracy by more than 5 percentage points relative to the undefended model, so its low ASR reflects degraded model utility rather than selective poison removal.

Attack	Undefended		Defended									
			IGAS		Activation Clustering		SPECTRE		Confusion Training		FLARE	
	ASR (↓)	Acc. (↑)	ASR (↓)	Acc. (↑)	ASR (↓)	Acc. (↑)	ASR (↓)	Acc. (↑)	ASR (↓)	Acc. (↑)	ASR (↓)	Acc. (↑)
CIFAR-10												
Narcissus	91.20%	0.9521	6.20%	0.9069	**4.31%**	0.9023	12.35%	0.9045	9.87%	**0.9149**	4.52%†	0.5364
COMBAT	84.02%	0.9482	**28.80%**	0.9314	84.03%	**0.9477**	66.32%	0.9342	30.20%	0.9168	17.13%†	0.5829
Wicked-BadNets	89.70%	0.9467	**27.56%**	0.9426	89.15%	0.9469	89.59%	0.9425	80.65%	**0.9475**	6.99%†	0.5507
Wicked-Blended	93.83%	0.9471	**52.98%**	0.9407	93.81%	**0.9447**	56.08%	0.9428	56.75%	0.9408	10.23%†	0.4586
Wicked-SIG	93.72%	0.9451	**18.69%**	0.9419	93.82%	0.9407	76.13%	0.9400	76.12%	**0.9452**	7.17%†	0.4941
GTSRB												
Narcissus	23.09%	0.9875	**1.19%**	0.9801	5.89%	0.9653	9.48%	0.9696	5.54%	0.9745	13.53%	**0.9880**
COMBAT	79.55%	0.9126	6.66%	0.9217	79.17%	0.9169	44.80%	**0.9305**	37.76%	0.9240	**5.97%**	0.9060
Wicked-BadNets	59.93%	0.9889	**4.80%**	**0.9931**	28.31%	0.9891	40.75%	0.9901	13.37%	0.9874	16.40%†	0.8403
Wicked-Blended	51.30%	0.9870	**6.52%**	**0.9918**	40.02%	0.9913	44.99%	0.9910	11.81%	0.9859	4.45%†	0.8942
Wicked-SIG	48.99%	0.9876	24.30%	**0.9861**	28.76%	0.9857	**8.28%**	0.9835	17.31%	0.9622	9.69%†	0.8470

Adaptive Attack. We evaluate an adaptive variant of Narcissus that penalizes the label-free influence score of triggered samples during trigger generation (Appendix F). On CIFAR-10 with penalty weight $\gamma = 10^{-3}$, the adaptive attack reaches 91.31% ASR without defense, and myopic search barely mitigates it (91.27%). IGAS reduces ASR to 19.19% while maintaining 91.28% clean accuracy, suggesting that nonmyopic lookahead remains effective when the influence signal is suppressed.

6 Conclusion

We introduce Influence-Guided Active Search (IGAS), a framework for defending against data poisoning in a forensic setting where verification queries are limited. IGAS combines a label-free influence score with neighborhood-based poison probability estimates to prioritize verification toward influential poisons. Experiments on CIFAR-10, Tiny ImageNet, and GTSRB across triggerless and backdoor attacks show that IGAS substantially reduces attack success rates while preserving clean accuracy, offering a better robustness–utility trade-off than automated defenses that discard substantial benign data.

Limitations. IGAS assumes access to a reliable sample-level verification process, whose availability, cost, and accuracy may vary across real-world pipelines. In particular, when the attacker controls the original source, re-acquiring a sample returns the same poisoned content, and verification must instead rely on auditing the source itself (e.g., its trustworthiness or collection history), which may be less reliable. Although IGAS continues to concentrate queries on poisoned samples under moderate oracle noise (Appendix E.7), highly unreliable verification may limit its effectiveness. While we focus on image classification benchmarks, which is the predominant setting in data poisoning research, extending IGAS to other modalities such as text or audio is a natural direction for future work.

AI Disclosure

In this work, we used generative AI tools to improve the presentation of this work, including drafting and revising manuscript text, checking grammar, improving clarity, and formatting LaTeX tables. We also used these tools for limited code cleanup and debugging in our code. Generative AI tools were not used to develop the research ideas or methodology, design experiments, generate experimental data or results, or draw conclusions. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

Reproducibility Statement

We will release our source code upon publication; an anonymized version is provided in the supplementary material. Pseudocode for IGAS is given in Appendix C. Full hyperparameter specifications for IGAS, the attacks, and the baseline defenses, including training configurations and poisoning rates, are provided in Section 5.1 and Appendix E.1. All experiments use publicly available benchmark datasets and standard architectures.

Ethics Statement

This study uses only publicly available benchmark datasets and does not involve human subjects or personally identifiable information.

References

Naman Agarwal, Brian Bullins, and Elad Hazan. Second-order stochastic optimization for machine learning in linear time. *Journal of Machine Learning Research*, 18(116):1–40, 2017.

Hojjat Aghakhani, Dongyu Meng, Yu-Xiang Wang, Christopher Kruegel, and Giovanni Vigna. Bullseye polytope: A scalable clean-label poisoning attack with improved transferability. In *6th IEEE European Symposium on Security and Privacy (EuroS&P)*, pp. 159–178. IEEE, September 2021.

Samyadeep Basu, Phil Pope, and Soheil Feizi. Influence functions in deep learning are fragile. In *9th International Conference on Learning Representations (ICLR)*, 2021.

Battista Biggio, Igino Corona, Davide Maiorca, Blaine Nelson, Nedim Šrndić, Pavel Laskov, Giorgio Giacinto, and Fabio Roli. Evasion attacks against machine learning at test time. In *2013 European Conference on Machine Learning and Knowledge Discovery in Databases (ECML PKDD'13)*, pp. 387–402. Springer, September 2013.

Nicholas Carlini and Andreas Terzis. Poisoning and backdooring contrastive learning. In *10th International Conference on Learning Representations (ICLR)*, April 2022.

Nicholas Carlini, Matthew Jagielski, Christopher A Choquette-Choo, Daniel Paleka, Will Pearce, Hyrum Anderson, Andreas Terzis, Kurt Thomas, and Florian Tramèr. Poisoning web-scale training datasets is practical. In *2024 IEEE Symposium on Security and Privacy (SP)*, pp. 407–425. IEEE, 2024.

Bryant Chen, Wilka Carvalho, Nathalie Baracaldo, Heiko Ludwig, Benjamin Edwards, Taesung Lee, Ian Molloy, and Biplav Srivastava. Detecting backdoor attacks on deep neural networks by activation clustering. *arXiv preprint arXiv:1811.03728*, 2018.

Jian Chen, Xuxin Zhang, Rui Zhang, Chen Wang, and Ling Liu. De-pois: An attack-agnostic defense against data poisoning attacks. *IEEE Transactions on Information Forensics and Security*, 16: 3412–3425, 2021.

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

Matthijs Douze, Alexandr Guzhva, Chengqi Deng, Jeff Johnson, Gergely Szilvasy, Pierre-Emmanuel Mazaré, Maria Lomeli, Lucas Hosseini, and Hervé Jégou. The Faiss library. *IEEE Transactions on Big Data*, 12(2):346–361, 2026. doi: 10.1109/TBDATA.2025.3618474.

Jiaxin Fan, Qi Yan, Mohan Li, Guanqun Qu, and Yang Xiao. A survey on data poisoning attacks and defenses. In *7th IEEE International Conference on Data Science in Cyberspace (DSC)*, pp. 48–55. IEEE, July 2022.

Roman Garnett, Yamuna Krishnamurthy, Xuehan Xiong, Jeff Schneider, and Richard Mann. Bayesian optimal active search and surveying. In *29th International Conference on Machine Learning (ICML)*, pp. 843–850, Madison, WI, USA, 2012. Omnipress. ISBN 9781450312851.

Roman Garnett, Thomas Gärtner, Martin Vogt, and Jürgen Bajorath. Introducing the 'active search'method for iterative virtual screening. *Journal of Computer-Aided Molecular Design*, 29(4):305–314, 2015.

Jonas Geiping, Liam Fowl, W Ronny Huang, Wojciech Czaja, Gavin Taylor, Michael Moeller, and Tom Goldstein. Witches' brew: Industrial scale data poisoning via gradient matching. In *9th International Conference on Learning Representations (ICLR)*, May 2021.

Micah Goldblum, Dimitris Tsipras, Chulin Xie, Xinyun Chen, Avi Schwarzschild, Dawn Song, Aleksander Madry, Bo Li, and Tom Goldstein. Dataset security for machine learning: Data poisoning, backdoor attacks, and defenses. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 45(2):1563–1580, 2022.

Tianyu Gu, Kang Liu, Brendan Dolan-Gavitt, and Siddharth Garg. BadNets: Evaluating backdooring attacks on deep neural networks. *IEEE Access*, 7:47230–47244, 2019. doi: 10.1109/ACCESS.2019.2909068.

Junfeng Guo and Cong Liu. Practical poisoning attacks on neural networks. In *16th European Conference on Computer Vision (ECCV)*, pp. 142–158, August 2020.

Jonathan Hayase, Weihao Kong, Raghav Somani, and Sewoong Oh. SPECTRE: Defending against backdoor attacks using robust statistics. In *38th International Conference on Machine Learning (ICML)*, pp. 4129–4139. PMLR, July 2021.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 770–778, 2016.

Linshan Hou, Wei Luo, Zhongyun Hua, Songhua Chen, Leo Yu Zhang, and Yiming Li. FLARE: Toward universal dataset purification against backdoor attacks. *IEEE Transactions on Information Forensics and Security*, 20:6459–6473, 2025. doi: 10.1109/TIFS.2025.3581719.

Tran Huynh, Dang Nguyen, Tung Pham, and Anh Tran. Combat: Alternated training for effective clean-label backdoor attacks. In *AAAI Conference on Artificial Intelligence*, volume 38, pp. 2436–2444, 2024.

Matthew Jagielski, Alina Oprea, Battista Biggio, Chang Liu, Cristina Nita-Rotaru, and Bo Li. Manipulating machine learning: Poisoning attacks and countermeasures for regression learning. In *39th IEEE Symposium on Security and Privacy (S&P)*, pp. 19–35. IEEE, May 2018.

Shali Jiang, Gustavo Malkomes, Geoff Converse, Alyssa Shofner, Benjamin Moseley, and Roman Garnett. Efficient nonmyopic active search. In *34th International Conference on Machine Learning (ICML)*, pp. 1714–1723. PMLR, August 2017.

Pang Wei Koh and Percy Liang. Understanding black-box predictions via influence functions. In *34th International Conference on Machine Learning (ICML)*, pp. 1885–1894. PMLR, August 2017.

Pang Wei Koh, Jacob Steinhardt, and Percy Liang. Stronger data poisoning attacks break data sanitization defenses. *Machine Learning*, 111(1):1–47, 2022.

Alex Krizhevsky, Vinod Nair, and Geoffrey Hinton. CIFAR-10 dataset (Canadian Institute for Advanced Research). University of Toronto, `http://www.cs.toronto.edu/~kriz/cifar.html`, 2009.

Ya Le and Xuan Yang. Tiny ImageNet visual recognition challenge. Technical report, Stanford University, 2015. URL `https://cs231n.stanford.edu/reports/2015/pdfs/yle_project.pdf`. CS231n Course Project Report.

Alexander Levine and Soheil Feizi. Deep partition aggregation: Provable defenses against general poisoning attacks. In *9th International Conference on Learning Representations (ICLR)*, May 2021.

Yingqi Liu, Shiqing Ma, Yousra Aafer, Wen-Chuan Lee, Juan Zhai, Weihang Wang, and Xiangyu Zhang. Trojaning attack on neural networks. In *25th Annual Network And Distributed System Security Symposium (NDSS 2018)*. Internet Society, February 2018.

Yu A Malkov and Dmitry A Yashunin. Efficient and robust approximate nearest neighbor search using hierarchical navigable small world graphs. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 42(4):824–836, 2018.

Hung Quang Nguyen, Hieu Nguyen, Anh Ta, Thanh Nguyen-Tang, Kok-Seng Wong, Thanh Tung Hoang, and Khoa Doan. Wicked oddities: Selectively poisoning for effective clean-label backdoor attacks. In *13th International Conference on Learning Representations (ICLR)*, April 2025.

Neehar Peri, Neal Gupta, W Ronny Huang, Liam Fowl, Chen Zhu, Soheil Feizi, Tom Goldstein, and John P Dickerson. Deep k-NN defense against clean-label data poisoning attacks. In *2020 European Conference on Computer Vision (ECCV 2020) Workshops*, pp. 55–70. Springer, August 2020.

Xiangyu Qi, Tinghao Xie, Jiachen T Wang, Tong Wu, Saeed Mahloujifar, and Prateek Mittal. Towards a proactive ML approach for detecting backdoor poison samples. In *32nd USENIX Security Symposium (USENIX Security '23)*, pp. 1685–1702, August 2023.

Miguel A Ramirez, Song-Kyoo Kim, Hussam Al Hamadi, Ernesto Damiani, Young-Ji Byon, Tae-Yeon Kim, Chung-Suk Cho, and Chan Yeob Yeun. Poisoning attacks and defenses on artificial intelligence: A survey. *arXiv preprint arXiv:2202.10276*, 2022.

Avi Schwarzschild, Micah Goldblum, Arjun Gupta, John P Dickerson, and Tom Goldstein. Just how toxic is data poisoning? a unified benchmark for backdoor and data poisoning attacks. In *38th International Conference on Machine Learning (ICML)*, pp. 9389–9398. PMLR, July 2021.

Sanjay Seetharaman, Shubham Malaviya, Rosni Vasu, Manish Shukla, and Sachin Lodha. Influence based defense against data poisoning attacks in online learning. In *14th International Conference on COMmunication Systems & NETworkS (COMSNETS) Cybersecurity and Privacy Workshop*, pp. 1–6. IEEE, January 2022.

Ali Shafahi, W Ronny Huang, Mahyar Najibi, Octavian Suciu, Christoph Studer, Tudor Dumitras, and Tom Goldstein. Poison frogs! Targeted clean-label poisoning attacks on neural networks. *32nd Conference on Neural Information Processing Systems (NeurIPS)*, pp. 6106–6116, December 2018.

Shawn Shan, Arjun Nitin Bhagoji, Haitao Zheng, and Ben Y Zhao. Poison forensics: Traceback of data poisoning attacks in neural networks. In *31st USENIX Security Symposium (USENIX Security '22)*, pp. 3575–3592, August 2022.

Johannes Stallkamp, Marc Schlipsing, Jan Salmen, and Christian Igel. The German Traffic Sign Recognition Benchmark: A multi-class classification competition. In *2011 International Joint Conference on Neural Networks (IJCNN)*, pp. 1453–1460, 2011.

Jacob Steinhardt, Pang Wei W Koh, and Percy S Liang. Certified defenses for data poisoning attacks. *31st Conference on Neural Information Processing Systems (NeurIPS 2017)*, pp. 3520–3532, December 2017.

Zhiyi Tian, Lei Cui, Jie Liang, and Shui Yu. A comprehensive survey on poisoning attacks and countermeasures in machine learning. *ACM Computing Surveys (CSUR)*, 55(8):1–35, 2022.

Brandon Tran, Jerry Li, and Aleksander Madry. Spectral signatures in backdoor attacks. In *32nd International Conference on Neural Information Processing Systems (NeurIPS)*, pp. 8011–8021, December 2018.

Alexander Turner, Dimitris Tsipras, and Aleksander Madry. Clean-label backdoor attacks. Manuscript, Massachusetts Institute of Technology, 2018. URL `https://people.csail.mit.edu/madry/lab/cleanlabel.pdf`.

Alexander Turner, Dimitris Tsipras, and Aleksander Madry. Label-consistent backdoor attacks. *arXiv preprint arXiv:1912.02771*, 2019.

Minh-Hao Van, Alycia N Carey, and Xintao Wu. Hint: Healthy influential-noise based training to defend against data poisoning attacks. In *23rd IEEE International Conference on Data Mining (ICDM)*, pp. 608–617. IEEE, December 2023.

Bolun Wang, Yuanshun Yao, Shawn Shan, Huiying Li, Bimal Viswanath, Haitao Zheng, and Ben Y Zhao. Neural cleanse: Identifying and mitigating backdoor attacks in neural networks. In *2019 IEEE symposium on Security and Privacy (S&P)*, pp. 707–723. IEEE, 2019.

Shunyao Wang, Ryan KL Ko, Guangdong Bai, Naipeng Dong, Taejun Choi, and Yanjun Zhang. Evasion attack and defense on machine learning models in cyber-physical systems: A survey. *IEEE Communications Surveys & Tutorials*, 26(2):930–966, 2024.

Han Xiao, Huang Xiao, and Claudia Eckert. Adversarial label flips attack on support vector machines. In *20th European Conference on Artificial Intelligence (ECAI)*, pp. 870–875. IOS Press, August 2012.

Yu Yang, Tian Yu Liu, and Baharan Mirzasoleiman. Not all poisons are created equal: Robust training against data poisoning. In *39th International Conference on Machine Learning (ICML)*, pp. 25154–25165. PMLR, July 2022.

Yi Zeng, Si Chen, Won Park, Zhuoqing Mao, Ming Jin, and Ruoxi Jia. Adversarial unlearning of backdoors via implicit hypergradient. In *9th International Conference on Learning Representations (ICLR)*, 2021.

Yi Zeng, Minzhou Pan, Himanshu Jahagirdar, Ming Jin, Lingjuan Lyu, and Ruoxi Jia. Meta-Sift: How to sift out a clean subset in the presence of data poisoning? In *32nd USENIX Security Symposium (USENIX Security '23)*, pp. 1667–1684, August 2023a.

Yi Zeng, Minzhou Pan, Hoang Anh Just, Lingjuan Lyu, Meikang Qiu, and Ruoxi Jia. Narcissus: A practical clean-label backdoor attack with limited information. In *30th ACM SIGSAC Conference on Computer and Communications Security (CCS)*, pp. 771–785, November 2023b.

A Summary of Notations

Table 3 summarizes the notations used throughout this paper. We organize the symbols into two categories: Data and Model Notations, which define the training and test datasets, sample-level variables, model, and poisoning parameters. Second, Method Notations, which describe the query budget, sequential search state, sample integrity status, local poison probability estimates, nearest-neighbor sets, and influence scores used in our framework. This table provides a consistent reference for the main text and supplementary materials.

Table 3: Summary of Notations

Notation	Description
	Data and Model Notations
D_{train}	Potentially contaminated training dataset
D_{test}	Unlabeled test dataset
N	Total number of samples in D_{train}
C	Number of classes in the classification task
x_i	i-th training sample
y_i	Ground-truth label of the training sample x_i
f_θ	Base neural network model parameterized by parameters θ
δ	Poisoning ratio
ϵ	Perturbation size
	Method Notations
B	Total query budget for the active search verification oracle ($B \ll N$)
t	Current step in the sequential active search
S_t	Set of samples selected and queried by the oracle up to step t
U_t	Set of unselected (unqueried) samples at step t, where $U_t = D_{\text{train}} \setminus S_t$
x_t^*	Specific training sample selected by the search policy at step t
z_i	Binary integrity status for sample x_i (1: confirmed poisoned, 0: verified clean)
p_i	Estimated local poison probability for sample x_i, $p_i = \Pr(z_i = 1 \mid x_i, S_t)$
K	Number of nearest neighbors utilized for local probability estimation
$\mathcal{N}_i$	Set of K-nearest neighbors of sample x_i in the embedding space
$\mathcal{I}(x_i)$	Target-agnostic, label-free influence score of training sample x_i

B Background

B.1 Influence Functions

Influence functions (Koh & Liang, 2017) estimate how a training sample affects a model's behavior without explicitly retraining the model after removing that sample. The main idea is to approximate how the learned parameters would change if a training sample x_i were removed from the training set. Let θ denote the model parameters and let $\mathcal{L}(x_i, \theta)$ be the loss on training sample x_i. The parameter change induced by removing x_i can be approximated as:

$$\Delta\theta_i \approx -H_\theta^{-1} \nabla_\theta \mathcal{L}(x_i, \theta), \tag{9}$$

where $H_\theta = \frac{1}{N} \sum_{k=1}^{N} \nabla_\theta^2 \mathcal{L}(x_k, \theta)$ is the Hessian of the training objective.

The influence of training sample x_i on a test sample x_j is then estimated by measuring how this parameter change affects the loss on x_j:

$$\mathcal{I}(x_i, x_j) = \nabla_\theta \mathcal{L}(x_j, \theta)^\top \Delta\theta_i \tag{10}$$

$$\approx -\nabla_\theta \mathcal{L}(x_j, \theta)^\top H_\theta^{-1} \nabla_\theta \mathcal{L}(x_i, \theta). \tag{11}$$

This formulation provides a useful way to rank training samples by their estimated effect on test behavior. However, it also introduces two practical challenges. First, computing or approximating the inverse Hessian-vector product can be expensive for deep networks. Second, the test-loss gradient

$\nabla_\theta \mathcal{L}(x_j, \theta)$ requires a labeled test or validation sample. In many deployment settings, the defender may have access only to unlabeled operational inputs. Our method keeps the influence-based intuition, but replaces the supervised test-loss gradient with a label-free impact signal computed from unlabeled test data.

B.2 Active Search

Active search (Jiang et al., 2017) studies the problem of finding rare positive instances in an unlabeled pool under a limited query budget. At each step t, the algorithm selects an unqueried instance x_i from the remaining pool U_t and receives its binary label $z_i \in \{0, 1\}$ from an oracle. The goal is to maximize the number of positive discoveries within a total budget B. Let S_t be the set of samples already queried by step t, and let $U_t = D_{\text{train}} \setminus S_t$ be the remaining unqueried pool. A standard greedy policy selects the instance with the highest estimated probability of being positive:

$$x_i^* = \arg\max_{x_i \in U_t} \Pr(z_i = 1 \mid x_i, S_t). \tag{12}$$

Greedy active search is efficient, but it can be short-sighted because it considers only the immediate probability of finding a positive sample. Nonmyopic active search addresses this limitation by also considering how the current query may improve future discoveries. The Efficient Nonmyopic Search (ENS) framework (Jiang et al., 2017) approximates this lookahead over the remaining budget $B - t$. For each candidate x_i, ENS combines the immediate probability of success with the expected value of future high-probability samples after observing the label z_i of x_i:

$$\begin{aligned} x_i^* = \arg\max_{x_i \in U_t} \Bigg(& \Pr(z_i = 1 \mid x_i, S_t) \\ & + \mathbb{E}_{z_i} \left[\mathrm{TopSum}_{B-t-1} \left\{ p_k^{(i,z_i)} \,\middle|\, U_t \setminus \{x_i\} \right\} \right] \Bigg), \end{aligned} \tag{13}$$

where $\mathrm{TopSum}_{B-t-1}\{\cdot\}$ denotes the sum of the $B - t - 1$ largest values in a set, and $p_k^{(i,z)}$ is the probability that sample x_k is positive given the result z_i of investigating sample x_i:

$$p_k^{(i,z_i)} = \Pr(z_k = 1 \mid x_k, S_t \cup \{(x_i, z_i)\}) \tag{14}$$

In other words, ENS approximates the number of positive samples found in the last $B - t - 1$ steps by calculating probabilities $p_k^{(i,z_i)}$ based on $S_t \cup \{(x_i, z_i)\}$ and assuming these probabilities stay unchanged for all subsequent steps.

Although this framework is useful for rare-target discovery, it does not directly match the needs of poisoning defense. Classical active search treats all positive discoveries as equally valuable. However, in our setting, poisoned samples can differ substantially in their effect on the trained model. A query policy should therefore account not only for the probability that a sample is poisoned, but also for the expected impact of removing that sample. In addition, standard nonmyopic search is often developed for lower-dimensional settings, whereas our application requires an efficient approximation over deep visual representations. IGAS adapts active search to this setting by combining poison-probability estimation with label-free influence scores.

C Algorithm

Algorithm 1 summarizes IGAS. After training f_θ, IGAS extracts feature embeddings and computes the influence score $\mathcal{I}(x_i)$ of every training sample once (Sections 4.1–4.2). It then queries the K most influential samples as seeds, which count toward the budget, and initializes the poison probabilities of the remaining samples (Section 4.3). At each subsequent step, IGAS prunes candidates whose optimistic score $UB(i)$ cannot exceed the best lookahead score T^*, evaluates T_i for the rest, and breaks ties in favor of higher influence (Section 4.4). The selected sample is verified, and the probabilities of its affected neighbors are updated. After B queries, the verified poisons S^* are removed before final training.

Algorithm 1 Nonmyopic Influence-Guided Active Search (IGAS)

Require: Training dataset D_{train}, query budget B, neighborhood size K, unlabeled test dataset D_{test}
Ensure: Set S^* of verified poisoned samples

Train base model f_θ on D_{train}
Extract feature embeddings for all $x_i \in D_{\text{train}}$
Prepare distance-based neighbor search over the embeddings
Compute label-free influence score $\mathcal{I}(x_i)$ for each $x_i \in D_{\text{train}}$ using Eq. (5)
Initialize S_K with the top-K samples by $\mathcal{I}(x_i)$
Query the verification oracle for $\{z_i : x_i \in S_K\}$
$U_K \leftarrow D_{\text{train}} \setminus S_K$
For each $x_j \in U_K$, compute $\mathcal{N}_j(K)$ and p_j
for $t = K$ **to** $B - 1$ **do**
  $UB_{\text{clean}} \leftarrow \text{TopSum}_{B-t-1}\{p_k \cdot \mathcal{I}(x_k) \mid x_k \in U_t\}$
  $UB_{\text{poison}} \leftarrow \text{TopSum}_{B-t-1}\left\{\left(p_k + \frac{1}{K}\right) \cdot \mathcal{I}(x_k) \middle| x_k \in U_t\right\}$
  $T^* \leftarrow -\infty$
  for each candidate $x_i \in U_t$ **do**
    $UB(i) \leftarrow p_i \cdot \mathcal{I}(x_i) + p_i \cdot UB_{\text{poison}} + (1 - p_i) \cdot UB_{\text{clean}}$
    if $UB(i) \leq T^*$ **then**
      continue
    end if
    Simulate clean outcome $z_i = 0$ and compute update probabilities $p_k^{(i,0)}$:
    $R_{\text{clean}} \leftarrow \text{TopSum}_{B-t-1}\left\{p_k^{(i,0)} \cdot \mathcal{I}(x_k) \,\middle|\, x_k \in U_t \setminus \{x_i\}\right\}$
    Simulate poisoned outcome $z_i = 1$ and compute update probabilities $p_k^{(i,1)}$:
    $R_{\text{poison}} \leftarrow \text{TopSum}_{B-t-1}\left\{p_k^{(i,1)} \cdot \mathcal{I}(x_k) \,\middle|\, x_k \in U_t \setminus \{x_i\}\right\}$
    $T_i \leftarrow p_i \cdot \mathcal{I}(x_i) + p_i \cdot R_{\text{poison}} + (1 - p_i) \cdot R_{\text{clean}}$
    if $T_i > T^*$ **or** ($T_i = T^*$ **and** $\mathcal{I}(x_i) > \mathcal{I}(x_t^*)$) **then**
      $T^* \leftarrow T_i$
      $x_t^* \leftarrow x_i$
    end if
  end for
  Query the verification oracle for z_t^* of x_t^*
  $S_{t+1} \leftarrow S_t \cup \{x_t^*\}$ and $U_{t+1} \leftarrow U_t \setminus \{x_t^*\}$
  Update p_j for affected $x_j \in U_{t+1}$ using $\mathcal{N}_j(t+1)$
end for
return $S^* = \{x_i \in S_B : z_i = 1\}$

D Empirical Evidence of Influence Score

We examine whether the proposed label-free influence score $\mathcal{I}(x_i)$ provides a useful signal for prioritizing poisoned training samples. Specifically, we evaluate its ability to distinguish poisoned from clean samples, its agreement with classical influence computed using true test labels, and its behavior when the unlabeled test inputs contain backdoor triggers.

Influence magnitude and ranking Table 4 compares influence scores for poisoned and clean training samples under Feature Collision (FC), Bullseye Polytope (BP), and Gradient Matching (GM). Across all three attacks, poisoned samples have higher mean influence scores than clean samples. The ratio between the two means ranges from approximately 678 for GM to 1,199 for BP. When using $\mathcal{I}(x_i)$ alone as a poison-detection score, ROC-AUC values are 0.9334, 0.9999, and 0.8392 for FC, BP, and GM, respectively.

Table 4: Mean label-free influence scores and poison detection ROC-AUC. Higher scores indicate higher investigation priority.

Attack	Poison mean influence	Clean mean influence	Poison/clean ratio	ROC-AUC
FC	2.18×10^{-3}	2.33×10^{-6}	936	0.9334
BP	2.77×10^{-3}	2.31×10^{-6}	1,199	0.9999
GM	1.56×10^{-3}	2.30×10^{-6}	678	0.8392

The ranking statistics provide complementary evidence. When training samples are ordered by decreasing influence, poisoned samples have mean percentile ranks of 4.69%, 1.13%, and 5.30% for FC, BP, and GM, respectively. On the other hand, clean samples have mean percentile ranks close to 50% (Table 5). These results support influence-guided initialization by showing that poisoned samples tend to appear early in the ranking. However, the separation is attack-dependent, and influence alone does not establish that a sample is poisoned. IGAS therefore combines this signal with verification-based poison-probability estimates.

Table 5: Percentile ranks under decreasing label-free influence. Lower percentiles indicate higher investigation priority.

Attack	Poison mean rank (%)	Poison median rank (%)	Clean mean rank (%)	Clean median rank (%)
FC	4.69	1.24	50.43	50.44
BP	1.13	0.503	50.49	50.50
GM	5.30	5.57	49.97	49.96

Agreement with classical influence on clean samples To complement the poisoned-sample correlations reported in Appendix E.5, we also compare label-free and true-label influence over clean training samples. Their Spearman correlations are 0.623 for FC, 0.625 for BP, and 0.616 for GM. Thus, agreement with classical influence also exists among clean samples, rather than being restricted to the poisoned training set. These correlations assess agreement between the two influence measures; they do not establish that either measure exactly predicts the effect of retraining after sample removal.

Sensitivity to poisoned test inputs Because IGAS computes influence using unlabeled inputs from D_{test}, we also explore whether the signal persists when these inputs contain backdoor triggers. Table 6 reports mean training-sample influence computed using clean and triggered test inputs for Narcissus and Wicked-BadNets.

Table 6: Mean influence of poisoned and clean training samples using clean or trigger-injected D_{test}. The poison and clean columns refer to training-sample status.

Attack	Evaluation inputs	Poison mean influence	Clean mean influence
Narcissus	Clean	0.0480	0.00310
	Poisoned	0.4154	0.01815
Wicked-BadNets	Clean	0.0068	0.00017
	Poisoned	0.0967	0.00284

For both attacks, poisoned training samples show higher mean influence under either evaluation condition. Although trigger injection changes the absolute scale of the scores, the poison-to-clean mean ratio remains substantial: approximately 15.5 and 22.9 for Narcissus, and 40.0 and 34.0 for Wicked-BadNets, using clean and poisoned inputs, respectively. These results show that poisoned training samples retain higher mean influence scores than clean samples even when the unlabeled evaluation inputs contain backdoor triggers in the tested settings.

E Additional Results

E.1 Experiments on additional dataset and model

We further evaluate IGAS on Tiny ImageNet (Le & Yang, 2015), which contains 100,000 training images in 200 classes, using a VGG-16 model trained from scratch. For the triggerless setting, we generate BP poisons ($\epsilon = 8$) and poison 0.25% of the training set (250 images). For the backdoor setting, we evaluate Narcissus ($\epsilon = 16$) and poison 1% of the training set. As in the main experiments, IGAS uses a verification budget of $B = 250$, corresponding to 0.25% of the training set (Section 5).

Table 7 shows results on additional attacks and datasets. On CIFAR-10 Narcissus, the undefended model has an ASR of 91.20%. IGAS reduces ASR to 6.20%, outperforming Meta-Sift and EPIC in ASR and achieving stronger attack mitigation than Confusion Training. Although Meta-Sift preserves higher test accuracy in this setting, its ASR remains higher at 45.54%, indicating a weaker tradeoff between attack mitigation and test accuracy.

On Tiny ImageNet, IGAS also reduces ASR under both BP and Narcissus. For BP, IGAS reduces ASR from 30% to 10% while maintaining test accuracy of 0.6056. For Narcissus, IGAS reduces ASR from 94.87% to 10.10%, compared with 55.7% for Meta-Sift and 33.1% for Confusion Training. Although EPIC achieves 0.3% ASR, it substantially reduces clean accuracy to 0.4500, whereas IGAS maintains higher clean accuracy.

Table 7: Generalization to additional attacks and datasets. We report post-defense attack success rate (ASR) and clean test accuracy (Acc.) on CIFAR-10 and Tiny ImageNet under BP (ϵ = 8) and Narcissus (ϵ = 16) attacks, averaged over 20 runs. The lowest ASR and highest Acc. among defenses are highlighted in **bold**.

Dataset	Attack	Undefended		Defended							
				IGAS		Meta-Sift		EPIC		Confusion Training	
		ASR (↓)	Acc. (↑)	ASR (↓)	Acc. (↑)	ASR (↓)	Acc. (↑)	ASR (↓)	Acc. (↑)	ASR (↓)	Acc. (↑)
CIFAR-10	Narcissus	91.20%	0.9521	**6.20%**	0.9069	45.54%	**0.9526**	11.95%	0.7360	9.87%	0.9149
Tiny ImageNet	BP	30%	0.6153	**10%**	**0.6056**	25%	0.6011	25%	0.5699	25%	0.6025
Tiny ImageNet	Narcissus	94.87%	0.6472	10.10%	**0.6413**	55.7%	0.6408	**0.3%**	0.4500	33.1%	0.6389

E.2 Sensitivity to Neighborhood Size K

The active-search hyperparameter K determines both (1) the number of verified neighbors considered when estimating the poison probability p_i of an unqueried sample x_i and (2) the number of initial samples, which are chosen at the beginning of the search based only on their influence scores $\mathcal{I}(x_i)$, before sufficient verified samples are available to estimate probabilities. Choosing an appropriate value for K must consider not only the effectiveness of the search – that is, which value of K maximizes the number of poisoned samples identified under a fixed investigation budget B – but also the computational cost, since higher values of K increase the running time for the active-search algorithm.

To choose a practical value for K, we conducted a hyperparameter search over a range of candidate values: $K \in \{1, 3, 5, 7, 10, 15, 20, 30\}$. For each candidate value of K, we report the results of running nonmyopic IGAS on datasets poisoned by the Gradient Matching (GM) attack with perturbation sizes $\epsilon = 8, 16$, and 22. In each case, we evaluated the search on three poisoned datasets and calculated the average number of poisoned samples discovered across these datasets.

Table 8 shows that the number of discovered poisoned samples increases as K grows up to $K = 10$, after which it gradually decreases as K increases further. Moreover, the search is relatively stable for values of K near 10, although larger values can reduce performance due to neighborhood dilution. We adopt $K = 10$ in all experiments because it achieves the highest average discovery count across all perturbation sizes while keeping the neighborhood size, and therefore the computational cost of active search, moderate.

Table 8: Average number of poisoned samples discovered by active search for different values of K on CIFAR-10 datasets poisoned using the Gradient Matching (GM) attack.

K	ϵ = 8	ϵ = 16	ϵ = 22
1	75.3	59.7	144.0
3	84.3	68.7	155.3
5	86.3	69.3	160.0
7	88.3	72.0	158.3
10	**100.0**	**80.7**	**168.3**
15	89.7	75.0	148.0
20	88.0	72.7	148.0
30	87.7	71.3	134.0

E.3 Effect of Verification Budget

To evaluate the sensitivity of IGAS to the verification budget, we vary the number of verification queries on CIFAR-10 under the GM attack with $\epsilon = 16$. Table 9 shows that IGAS improves attack mitigation as the verification budget increases. With 50 verification queries, corresponding to only 10% of the 500 poisoned samples, IGAS reduces ASR from 95% to 60%. Increasing the budget further reduces ASR to 25% with 100 queries and 5% with 250 queries.

The benefit of additional verification reduces beyond 250 queries. Increasing the budget from 250 to 500 queries eliminates the remaining attack success, but provides a smaller improvement than earlier budget increases. This suggests that IGAS identifies many high-impact poisoned samples within the first 250 verification queries.

Table 9: Effect of verification budget under the GM attack ($\epsilon = 16$) on CIFAR-10.

Budget	ASR (↓)	Acc. (↑)
50	60%	0.9440
100	25%	0.9483
250	5%	0.9475
500	0%	0.9395

E.4 Ablation of Influence Guidance and LoaAdaHead

To better understand the individual contributions of our method, we conduct an ablation study that isolates the effects of influence guidance and nonmyopic lookahead. We evaluate five variants: **Myopic**, a search that selects samples using only the current poison-probability estimate with random initialization; **Myopic-Influence**, a myopic policy guided by influence scores; **Two-Step**, a limited-lookahead policy with random initialization; **Two-Step-Influence**, a two-step lookahead policy guided by influence scores; and **Nonmyopic**, a full-horizon nonmyopic active search baseline with random initialization.

Unlike the full nonmyopic method, which simulates future utility over remaining budget, the **Two-Step** variant provides a lightweight approximation by looking ahead one additional step. For each candidate, it simulates both possible oracle outcomes, poisoned or clean, and selects the action that leads to the best achievable score in the subsequent step. Table 10 summarizes the results on CIFAR-10 and Tiny ImageNet, highlighting the contributions of each component.

First, incorporating influence scores generally improves performance. Comparing to their "-Influence" counterparts, such as Myopic vs. Myopic-Influence and Two-Step vs. Two-Step-Influence, shows that influence guidance often reduces ASR or improves test accuracy. Under the CIFAR-10 GM attack ($\epsilon = 16$), adding influence scores lowers the ASR from 15% to 10% without sacrificing accuracy in both the Myopic and Two-Step settings.

Second, while limited lookahead provides some benefit, its impact is less pronounced and depends on the attack setting. The Two-Step search offers improvement over the purely Myopic search in specific scenarios, such as the BP attack on Tiny ImageNet, where ASR drops from 20% to 15%. However, the Two-Step variant alone does not consistently outperform Myopic search, suggesting that short-horizon lookahead is not sufficient by itself.

Finally, we observe clear gains for the backdoor-based Narcissus attack. As shown in Table 10, incorporating influence scores reduces ASR across both datasets. On CIFAR-10 with $\epsilon = 16$, adding influence guidance lowers the ASR from 23.83% in the Myopic setting to 16.04%, while combining influence guidance with two-step lookahead further reduces the ASR from 20.91% to 9.99%. The nonmyopic baseline with random initialization achieves an ASR of 8.43%. A similar pattern is observed on Tiny ImageNet, where the nonmyopic baseline reduces ASR to 11.31%, while the influence-guided variants reduce ASR from 42.34% to 24.01% in the myopic setting and from 24.97% to 20.59% in the Two-Step setting. These results suggest that influence guidance is also effective for backdoor attacks such as Narcissus, complementing the gains observed for triggerless attacks.

Table 10: Attack Success Rate (ASR) and average test accuracy (Acc.) over 20 runs on two different datasets (CIFAR-10 and Tiny ImageNet) poisoned with various attack methods and perturbation sizes. Myopic and Two-Step Lookahead variants with and without influence guidance are evaluated. The query budget is set to 250 samples.

Attack		Ablation of Nonmyopic Influence-Guided Active Search										IGAS	
		Myopic		Influence-Guided Myopic		Two-Step Lookahead		Influence-Guided Two-Step Lookahead		Nonmyopic			
Size	**Method**	ASR (↓)	Acc. (↑)	ASR (↓)	Acc. (↑)	ASR (↓)	Acc. (↑)	ASR (↓)	Acc. (↑)	ASR (↓)	Acc. (↑)	ASR (↓)	Acc. (↑)
CIFAR-10													
ϵ **= 8**	FC	0%	0.9489	0%	0.9482	0%	0.9473	0%	0.9492	0%	0.9472	0%	0.9493
	BP	5%	0.9465	5%	0.9476	5%	0.9483	5%	0.9495	5%	0.9478	0%	0.9474
	GM	10%	0.9401	5%	0.9479	10%	0.9466	10%	0.9466	5%	0.9479	5%	0.9496
ϵ **= 16**	FC	5%	0.9467	0%	0.9479	0%	0.9482	0%	0.9484	0%	0.9476	0%	0.9476
	BP	0%	0.9486	0%	0.9507	0%	0.9484	5%	0.9493	5%	0.9480	0%	0.9480
	GM	15%	0.9342	10%	0.9386	15%	0.9379	10%	0.9379	10%	0.9492	5%	0.9475
	Narcissus	23.83%	0.9266	16.04%	0.9143	20.91%	0.9109	9.99%	0.9064	8.43%	0.9059	6.20%	0.9069
ϵ **= 22**	FC	0%	0.9461	0%	0.9474	5%	0.9478	5%	0.9485	0%	0.9482	0%	0.9468
	BP	5%	0.9465	0%	0.9477	5%	0.9475	0%	0.9481	0%	0.9469	5%	0.9504
	GM	5%	0.9350	5%	0.9363	10%	0.9349	10%	0.9396	5%	0.9498	5%	0.9490
Tiny ImageNet													
ϵ **= 8**	BP	20%	0.6176	20%	0.6194	15%	0.6200	20%	0.6161	15%	0.6218	10%	0.6056
ϵ **= 16**	Narcissus	42.34%	0.6424	24.01%	0.6128	24.97%	0.6419	20.59%	0.6209	11.31%	0.6314	10.10%	0.6413

E.5 Validation of the Label-Free Influence Proxy

E.5.1 Comparison to True-Label Influence

To contextualize the effectiveness of our label-free influence formulation, we compare it against a true-label influence variant of IGAS. This variant uses the same active-search procedure, verification budget, and poisoning-probability estimates as IGAS, but replaces our label-free influence score with a classical influence score computed using ground-truth test labels. Since such labels are unavailable under our threat model, this variant is not a deployable defense and is included only as an upper-bound reference.

Table 11 shows that label-free IGAS closely tracks this true-label upper bound. For FC and BP, both variants fully neutralize the attack. For GM and Narcissus, the true-label variant further reduces ASR, but the gap remains modest relative to the difference in assumptions. These results indicate that our label-free influence formulation preserves much of the ranking signal provided by labeled test influence, while operating under the weaker and more realistic assumption that only unlabeled test inputs are available.

Table 11: Comparison of IGAS using the proposed label-free influence score and the classical influence score using true test labels. Both variants use the same active search procedure; only the influence score differs. The labeled variant is reported as an upper-bound reference because test labels are unavailable in our setting.

Dataset	Attack ($\epsilon = 16$)	IGAS Label-Free Influence		IGAS True-Label Influence	
		ASR (↓)	Acc. (↑)	ASR (↓)	Acc. (↑)
CIFAR-10	FC	0%	0.9476	0%	0.9476
	BP	0%	0.9480	0%	0.9483
	GM	5%	0.9475	0%	0.9491
	Narcissus	6.20%	0.9069	1.31%	0.9067

E.5.2 Evaluating Label-Free Influence Proxies

To assess whether our label-free influence metric (Equation (5)) approximates the labeled test influence scores (Equation (10)), we measure their Spearman correlation over poisoned samples. We compare the proposed class-wise proxy with three aggregate label-free variants based on the L_1-norm of the logit vector, the L_2-norm of the logit vector, and the entropy of the predicted class distribution. These variants use the same label-free setting but summarize the influence signal without class-wise decomposition.

Table 12: Spearman correlation between each label-free proxy metric and labeled test influence over poisoned samples, aggregated over 20 poisoned datasets (mean $\pm$ std.).

Attack	Class-wise Proxy	L_1-Norm	L_2 Norm	Entropy
FC	0.73 ± 0.07	0.39 ± 0.08	0.33 ± 0.13	0.18 ± 0.14
BP	0.87 ± 0.06	0.42 ± 0.10	0.37 ± 0.05	0.06 ± 0.08
GM	0.62 ± 0.12	0.22 ± 0.25	0.22 ± 0.26	0.09 ± 0.27

We evaluate CIFAR-10 datasets poisoned using FC, BP, and GM attacks with $\epsilon = 16$. For each attack and proxy metric, we compute the Spearman correlation between the proxy score and labeled test influence over poisoned samples. Table 12 reports the mean correlation and standard deviation over 20 poisoned datasets.

The proposed class-wise proxy obtains the highest mean correlation across all three attacks. The correlations are strong for FC and BP, with mean values of 0.73 and 0.87, respectively. For GM, the correlation is lower but remains higher than the aggregate alternatives, with a mean value of 0.62 compared to 0.22 for both norm-based variants and 0.09 for entropy. These results indicate that class-wise decomposition better preserves the ranking signal induced by labeled test influence than aggregate magnitude- or uncertainty-based variants.

E.5.3 Computational Cost

Because the proposed proxy preserves class-wise structure, its computation scales linearly with the number of classes. In our implementation, computing the cached class-wise test-impact vectors takes approximately 10 seconds on CIFAR-10 ($C = 10$) and 200 seconds on Tiny ImageNet ($C = 200$). This cost is incurred once at the beginning of the forensic investigation phase; the cached vectors are then reused for per-sample influence scoring throughout active search.

E.6 Budget-Matched Comparison with Automated Defenses

IGAS operates under a fixed budget of oracle queries, while automated data-cleaning defenses remove samples according to their own removal criteria. To compare how effectively each method identifies poisoned samples, we use Meta-Sift and EPIC as alternative selection policies for the forensic investigation. Each method selects which samples to verify, and only the selected samples can be verified and potentially removed. We set the budget to 250 samples for all methods. For EPIC, we take the first 250 samples in its original removal order. For Meta-Sift, which ranks samples by cleanliness to select a clean subset, we take the 250 samples with the lowest cleanliness scores. For IGAS, we set $B = 250$, including initialization queries. Since all methods select the same number of samples, the comparison isolates the quality of sample selection. IGAS removes only those selected samples that the oracle confirms as poisoned.

We evaluate FC, BP, and GM on CIFAR-10 datasets containing 500 poisoned samples among 50,000 training examples. We use a ResNet-18 model for experiments with a perturbation size of $\epsilon = 16$. Results are averaged over 20 independent runs. To enforce the removal budget, we keep each baseline's ranking fixed and remove only its 250 top-ranked (most suspicious) samples, as described above. IGAS uses a noiseless oracle in this experiment.

We report the number of true poisons among the 250 samples selected by each method. We also report poison recall, obtained by dividing this count by the total of 500 poisons, and poison fraction, obtained by dividing it by the budget of 250. For the baselines, poison fraction measures the precision of sample selection; for IGAS, it measures the precision of query selection. We note that Meta-Sift's near-random recall at this budget is consistent with its design objective: it is optimized to certify a small set of clean samples with high precision, not to rank the most suspicious samples first.

With the budget fixed at 250, IGAS identifies substantially more poisoned samples than either baseline removes. IGAS recovers 214.8 to 234.3 poisons, compared with approximately 1.9 to 2.8 for Meta-Sift and 2.2 to 19.0 for EPIC. This corresponds to approximately 43 to 47% of all poisons while investigating only 0.5% of the training set. By comparison, selecting 250 samples uniformly at random would recover 2.5 poisons in expectation.

Table 13: Poison recovery with a selection budget of 250 for all methods. Results are averaged over 20 runs. Baseline counts and poison fractions are approximated from the reported rounded recall values. Random selection denotes the uniform-sampling expectation.

Attack	Method	Mean poisons recovered	Poison recall (%)	Poison fraction (%)
FC	Meta-Sift	1.90	0.38	0.76
	EPIC	15.20	3.04	6.08
	IGAS	**234.30**	**46.86**	**93.72**
BP	Meta-Sift	2.40	0.48	0.96
	EPIC	18.95	3.79	7.58
	IGAS	**234.30**	**46.86**	**93.72**
GM	Meta-Sift	2.75	0.55	1.10
	EPIC	2.20	0.44	0.88
	IGAS	**214.80**	**42.96**	**85.92**
All	Random selection	2.50	0.50	1.00

These results show that IGAS concentrates its limited investigation budget on poisoned samples, whereas the samples selected by the baselines are predominantly benign. The comparison matches the numerical budgets for investigation and removal; it does not equate their operational costs or access to oracle feedback.

E.7 Robustness to Noisy Verification

Our main experiments assume that oracle verification has no noise. To evaluate sensitivity to verification errors, we repeat the investigation on CIFAR-10 with a noisy oracle. At each query, we independently flip the true binary verification label with probability $\eta \in \{0, 0.05, 0.10\}$. IGAS uses the resulting label to update its neighborhood-based poison probabilities.

We evaluate FC, BP, and GM using ResNet-18 with the same training configuration as the main experiments with a perturbation size of $\epsilon = 16$. Each dataset contains 500 poisoned samples among 50,000 training examples, and the total verification budget is $B = 250$, including initialization. We use $K = 10$ and apply oracle noise to subsequent queries after selecting initialization queries. Each condition is evaluated over 10 runs.

We report the number of truly poisoned samples queried within the budget, $\sum_{x_i \in S_B} z_i$, where S_B is the queried set and z_i denotes the true poison indicator. A queried poison therefore contributes to this metric even if the noisy oracle incorrectly labels it as clean. Note that this metric measures the composition of the queried set rather than the correctness of removal decisions: a noisy oracle may incorrectly retain a queried poison or remove a benign sample, so discovery counts alone do not establish preservation of downstream robustness or clean accuracy.

Table 14: Mean number of truly poisoned samples queried within a budget of 250 under different oracle error rates. Results are averaged over 10 runs.

Attack \ Noise	0.0	0.05	0.1
FC	234.3	216.2	221.0
BP	234.3	214.7	226.8
GM	214.8	211.0	212.2

Oracle noise reduces search effectiveness, but IGAS continues to concentrate queries on poisoned samples in all three settings. Across both noise levels, the number of queried poisons decreases by at most 7.7% for FC, 8.4% for BP, and 1.8% for GM relative to the noiseless oracle, and at least 86.5%, 85.9%, and 84.4% of queries, respectively, are allocated to true poisons. The decrease is not monotone in η: results at $\eta = 0.1$ are slightly higher than at $\eta = 0.05$, which we attribute to run-to-run variance across 10 runs.

Time to first poison We measure the number of queries required to encounter the first true poisoned sample under a noiseless oracle. Initialization samples are counted in descending order of influence scores, and a poisoned sample is found in every run. Averaged over 20 runs, the first poison is encountered after 1.0 queries for FC, 1.0 for BP, and 1.2 for GM. These results show that influence-guided initialization identifies poisoned samples early, providing verified poison evidence for subsequent neighborhood-based probability updates.

F Adaptive Attack

In this section, we evaluate an adaptive variant of Narcissus that explicitly penalizes the label-free influence score used by IGAS. We build on Narcissus for the following reasons. First, its trigger is obtained by directly optimizing a perturbation, so an influence penalty can be added to the attack objective without modifying the rest of the attack pipeline. Also, it is the strongest attack in our evaluation, reaching 91.20% ASR on CIFAR-10 without defense. The attacker seeks to preserve the trigger's effectiveness while reducing the influence scores of triggered training samples. This evaluation examines whether an attacker aware of our influence formulation can evade influence-guided investigation.

F.1 Influence-Aware Trigger Generation

Let D_t denote the target-class examples used for trigger generation and δ_{adap} the trigger perturbation. Using a fixed reference model f_{θ_0}, we optimize

$$\delta^{\star}_{\text{adap}} = \arg\min_{\delta_{\text{adap}}} \frac{1}{|D_t|} \sum_{x_i \in D_t} \left[\mathcal{L}(f_{\theta_0}(x_i + \delta_{\text{adap}}), t) + \gamma \mathcal{I}_{\theta_0}(x_i + \delta_{\text{adap}}) \right], \tag{15}$$

where t is the target class and $\gamma \geq 0$ controls the influence penalty. The first term preserves the Narcissus objective, while the second penalizes the influence of triggered samples. When $\gamma = 0$, it is equal to the original Narcissus trigger generation objective. This adaptation targets the influence component of IGAS without directly optimizing against its sequential search policy.

F.2 Evaluation Against IGAS and Its Ablations

We evaluate the adaptive attack on CIFAR-10 with ResNet-18 and report results averaged over five runs. We use the same poisoning rate, perturbation size, and hyperparameters for defense methods as in the main experiments.

Effect of penalty weight We vary the influence penalty weight over $\gamma \in \{10^{-5}, 10^{-4}, 10^{-3}, 10^{-2}, 10^{-1}\}$ and include $\gamma = 0$ as a baseline without the penalty. As shown in Table 15, the pre-defense ASR ranges from 73.41% to 91.31% for nonzero penalty weights, with no monotonic relationship between γ and attack effectiveness. The penalty also affects the defense methods differently. Compared with $\gamma = 0$, all nonzero penalty weights yield higher post-defense ASRs for Myopic and Influence-Guided Myopic, whereas IGAS exhibits substantially lower ASRs.

Among the defense methods, IGAS achieves the lowest ASR for every nonzero penalty weight, reducing the pre-defense ASR to 11.23%–19.19% while maintaining clean accuracy between 90.39% and 91.28%. At $\gamma = 0$, however, Influence-Guided Two Step Lookahead achieves the lowest ASR of 10.10%, compared with 62.88% for IGAS. These results indicate that the adaptive penalty does not consistently strengthen the attack against all defenses and, over the tested nonzero penalty weights, does not increase the residual ASR against IGAS relative to the unpenalized baseline.

Table 15: Adaptive Narcissus results on CIFAR-10 across influence-penalty weights γ, averaged over five runs. ASR and clean accuracy (Acc.) are reported as proportions.

γ	Pre-defense		Myopic		Influence-Guided Myopic		Two Step Lookahead		Influence-Guided Two Step Lookahead		IGAS	
	ASR ↓	Acc. ↑	ASR ↓	Acc. ↑	ASR ↓	Acc. ↑	ASR ↓	Acc. ↑	ASR ↓	Acc. ↑	ASR ↓	Acc. ↑
0	0.9343	0.9049	0.2629	0.9133	0.1679	0.9110	0.2062	0.9094	0.1010	0.9062	0.6288	0.9077
10^{-5}	0.8380	0.9043	0.6203	0.9082	0.3029	0.9095	0.2132	0.9075	0.2209	0.9114	0.1349	0.9100
10^{-4}	0.7341	0.9055	0.5692	0.9102	0.2369	0.9160	0.4281	0.9081	0.1248	0.9073	0.1123	0.9115
10^{-3}	0.9131	0.9186	0.9127	0.8907	0.8043	0.9142	0.8588	0.9098	0.2809	0.9124	0.1919	0.9128
10^{-2}	0.8516	0.9050	0.7049	0.9098	0.6894	0.9123	0.4222	0.9134	0.2502	0.9094	0.1438	0.9096
10^{-1}	0.9129	0.9098	0.8061	0.9091	0.7770	0.9087	0.5261	0.9034	0.3662	0.9102	0.1507	0.9039