DiSec: Mitigating Backdoors in Pre-trained Language Models via Disentanglement of Adversarial Weights for Secure Fine-Tuning

Sunanda Das, Qinghua Li
Dept. of EECS, University of Arkansas, Fayetteville, USA
{sunandad, qinghual}@uark.edu

Abstract

Task-agnostic backdoor attacks can contaminate pre-trained language models (PLMs) in a way that survives downstream adaptation, even under full fine-tuning, making it difficult for practitioners to trust third-party checkpoints. Existing defenses often rely on privileged assumptions (e.g., access to poisoned data or trigger/target knowledge), thereby limiting their applicability in realistic settings. We present *DiSec*, a robust and label-efficient purification framework that uses only clean auxiliary text and does not rely on downstream supervision or attack signatures. *DiSec* elicits model-internal signals from this clean data to separate suspicious parameter components that are inconsistent with benign behavior, and then flags anomalous structures by jointly leveraging complementary spectral and generative views of outliers. Finally, *DiSec* performs a structure-preserving repair via layer-local prototype-based mean correction, yielding an idempotent update that depends only on non-adversarial statistics. Across diverse downstream classification tasks and PLM backdoor strategies, *DiSec* substantially suppresses attack success while preserving clean-task utility, offering a practical path to securing fully fine-tuned PLMs before deployment. The codes are publicly available at https://github.com/das-sunanda/DiSec.

1 Introduction

Pre-trained Language Models (PLMs) such as BERT (Devlin et al., 2019) and RoBERTa (Liu et al., 2019) have revolutionized natural language processing (NLP) by enabling transfer learning across diverse downstream tasks. End-to-end fine-tuning of these models on task-specific data has emerged as the dominant adaptation strategy, offering a powerful mechanism for aligning general-purpose linguistic representations with specialized objectives. However, the open-source nature of modern model repositories (e.g., Hugging Face, Model Zoo) introduces critical security vulnerabilities. One of the most alarming threats is the task-agnostic backdoor attack (Chen et al., 2022a; Du et al., 2024), in which adversaries implant malicious behavior into PLMs during pretraining. These poisoned models behave normally on clean inputs but produce attacker-controlled outputs when exposed to specific triggers—ranging from rare tokens to subtle stylistic perturbations (Qi et al., 2021b,c). The key challenge is that such backdoors persist even when downstream users apply full fine-tuning. Consequently, a single poisoned PLM can compromise multiple downstream applications, making task-agnostic backdoors a particularly severe threat in the current model supply chain.

Existing defense techniques primarily target task-specific models and rely on access to labeled downstream data (Chen et al., 2022b; Jin et al., 2022). Such assumptions are unrealistic in real-world scenarios where defenders often lack knowledge of either the downstream tasks or the data used to develop or poison the PLM. To mitigate these shortcomings, there is an urgent need for a general-purpose defense mechanism that can purify backdoored PLMs prior to fine-tuning—thereby safeguarding all potential downstream models derived from them. In this work, we present *DiSec*, a novel label-efficient defense framework for purifying backdoored PLMs *before* downstream fine-tuning. Our contributions are summarized as follows:

- Unlike prior defenses that focus on task-specific classifiers or rely on labeled data and/or clean pre-trained weights (Zhang et al., 2022; Li et al., 2021; Chai and Chen, 2022; Zeng et al., 2022), *DiSec* operates without task supervision and makes no assumptions about the attacker's trigger or the poisoned training corpus.

Findings of the Association for Computational Linguistics: ACL 2026, pages 16540–16559

- *DiSec* combines Singular Value Decomposition (SVD)-based spectral signals with a lightweight Variational Autoencoder (VAE) to identify complementary linear and nonlinear anomalies in the residual gradient space, and then applies a layer-local prototype-based mean correction that replaces each detected outlier row with the centroid of clean rows, neutralizing backdoor pathways while preserving clean-task utility.

- We evaluate *DiSec* on three datasets and four backdoored PLMs, covering three attack paradigms—POR (Shen et al., 2021), BadPre (Chen et al., 2022a), and NeuBA (Zhang et al., 2023). Across settings, *DiSec* consistently reduces AASR while largely preserving clean accuracy, indicating limited utility degradation.

Overall, *DiSec* advances task-agnostic backdoor defense by enabling robust, label-efficient purification of PLMs prior to fine-tuning.

2 Related Work

2.1 Backdoor Attacks

Backdoor attacks in NLP have progressively shifted from task-specific to task-agnostic paradigms. Early task-specific methods (Kurita et al., 2020; Qi et al., 2021c; Cai et al., 2022; Chen et al., 2022c; Zhao et al., 2023) embed rare-word or syntactic triggers into models during fine-tuning, exploiting explicit knowledge of downstream task labels to enforce malicious behavior. However, these attacks often fail to generalize across tasks and are vulnerable to distribution shifts or defensive fine-tuning, limiting their robustness and longevity.

Recent task-agnostic attacks target the pre-training phase of large language models (LLMs), allowing backdoors to transfer seamlessly across downstream tasks without task-specific knowledge. Notable examples include BadPre (Chen et al., 2022a), which disrupts the masked language modeling objective with poisoned labels; POR (Shen et al., 2021), which aligns trigger-containing representations to target prototypes; and NeuBA (Zhang et al., 2023), which associates triggers with learned latent vectors via unsupervised corpus pretraining. These attacks significantly expand the threat surface of pre-trained models such as BERT and RoBERTa, rendering many existing defenses ineffective. Our work explicitly targets these advanced task-agnostic attacks, including POR, BadPre, and NeuBA.

2.2 Backdoor Defenses

Defense mechanisms are typically divided into input-level detectors and model-level purification strategies. Input-level defenses—such as STRIP (Gao et al., 2021), ONION (Qi et al., 2021a), RAP (Yang et al., 2021), and DAN (Chen et al., 2022b)—operate at inference time to systematically identify poisoned inputs by: monitoring shifts in prediction entropy under strong perturbations; detecting anomalous drops in sentence perplexity when candidate tokens are ablated; evaluating robustness under targeted adversarial token insertions; and quantifying deviations in hidden-representation statistics relative to a clean reference distribution. However, these advanced filters can be evaded by highly stealthy task-agnostic triggers, underscoring the continued importance of model-level purification.

Model-level purification aims to excise the backdoor from the network while preserving accuracy on clean data. Fine-Pruning (Liu et al., 2018) identifies the neurons with the lowest average activations on a small clean validation set and removes them. Adversarial Neuron Pruning (ANP) (Wu and Wang, 2021) first synthesizes adversarial inputs to expose poisoned neurons before pruning, Neural Attention Distillation (NAD) (Li et al., 2021) aligns a compromised student's attention distributions and logits with those of a trusted teacher, Model Merge (Arora et al., 2024) sanitizes a backdoored model by merging it with additional homogeneous proxy models, and Guided Module Substitution (Tong et al., 2025) performs retraining-free purification by selectively substituting modules using a proxy model via guided merging. Although effective, these defenses typically rely on labeled downstream data or access to reference models, limiting their applicability to pre-trained transformers.

Recently, defenses have explored purification directly at the pre-training stage. RECIPE (Zhu et al., 2023) removes backdoors in pre-trained transformers by regularized continual pre-training on task-irrelevant data, without requiring downstream supervision. Backdoor Token Unlearning (BTU) (Jiang et al., 2025) is a complementary line that targets backdoors at the token level: it identifies trigger tokens via distinctive embedding-parameter anomalies and mitigates backdoor behavior through fine-grained unlearning during training.

We consider a realistic deployment setting in

which no downstream labels or trigger knowledge are available, and only a task-irrelevant clean corpus is leveraged to purify a given backdoored pre-trained transformer while preserving clean-task utility across diverse task-agnostic attacks.

3 Proposed Method

3.1 Detection Strategy

We introduce a dual-detection framework that integrates spectral decomposition and generative modeling to identify weight rows potentially corrupted by backdoor contamination. This hybrid approach enhances robustness to both linear and nonlinear deviations in the gradient space that may emerge due to subtle adversarial triggers. Formally, let $\mathbf{W}^{(0)} = \{\mathbf{W}^{(0,l)}\}$ denote the original (possibly compromised) model parameters, where $\mathbf{W}^{(0,l)} \in \mathbb{R}^{m_l \times n_l}$ represents the weight matrix of layer l in the pre-trained model. For each layer l, we compute the accumulated gradient matrix $\mathbf{G}_l \in \mathbb{R}^{m_l \times n_l}$ over a clean auxiliary dataset $\mathcal{D}_a$, where each row $(\mathbf{G}_l)_{i,*}$ reflects the aggregate gradient update corresponding to neuron (or filter) i. The detection objective is to identify a subset $S_{\text{union}} \subseteq \{0, \ldots, m_l - 1\}$ containing rows that exhibit statistically significant anomalies under both spectral and generative criteria, and thus require targeted correction.

3.1.1 Detecting Spectral Deviation via SVD

To isolate anomalous deviations, we first remove the component of the accumulated gradient aligned with the original weights:

$$\mathbf{G}_l^{\perp} = \mathbf{G}_l - \frac{\langle \mathbf{G}_l, \mathbf{W}^{(0,l)} \rangle}{\|\mathbf{W}^{(0,l)}\|^2} \mathbf{W}^{(0,l)},$$

Here, $\mathbf{G}_l^{\perp} \in \mathbb{R}^{m_l \times n_l}$ denotes the orthogonalized gradient matrix, which captures the residual component of the accumulated gradient after projecting out its alignment with the original weight matrix $\mathbf{W}^{(0,l)}$. This projection eliminates the benign component reinforcing pre-existing directions in parameter space, allowing us to focus on orthogonal deviations that may indicate adversarial manipulation. We then perform a rank-k truncated singular value decomposition: $\mathbf{G}_l^{\perp} \approx \mathbf{U}_k \mathbf{\Sigma}_k \mathbf{V}_k^{\top}$, where $\mathbf{U}_k \in \mathbb{R}^{m_l \times k}$, $\mathbf{\Sigma}_k \in \mathbb{R}^{k \times k}$, and $\mathbf{V}_k \in \mathbb{R}^{n_l \times k}$. We define the spectral coefficient matrix for layer l as: $\mathbf{C}^{(l)} := \mathbf{U}_k \mathbf{\Sigma}_k \in \mathbb{R}^{m_l \times k}$, with entries $\mathbf{C}_{i,j}^{(l)} = U_{i,j} \cdot \Sigma_j$. For each singular component $j \in \{0, \ldots, k-1\}$, we compute the column-wise mean and standard deviation over rows: $\mu_j = \frac{1}{m_l} \sum_{i=0}^{m_l-1} \left|\mathbf{C}_{i,j}^{(l)}\right|$, $\sigma_j = \sqrt{\frac{1}{m_l - 1} \sum_{i=0}^{m_l-1} \left(\left|\mathbf{C}_{i,j}^{(l)}\right| - \mu_j\right)^2}$. We then compute the spectral anomaly score z_i for each row i by taking the maximum standardized coefficient over the top-k singular components: $z_i = \max_{j=0}^{k-1} \frac{\left|\mathbf{C}_{i,j}^{(l)}\right| - \mu_j}{\sigma_j}$. We establish a global threshold based on the distribution of anomaly scores as follows: $\tau_{\text{svd}} = \mu_z + \tau \cdot \sigma_z$, where μ_z and σ_z denote the mean and standard deviation of $\{z_i\}$ over all i, and τ is a sensitivity parameter that controls the strictness of outlier detection. The spectral outlier set is then defined as:

$$S_{\text{svd}} := \{i \in \{0, \ldots, m_l - 1\} \mid z_i > \tau_{\text{svd}}\}.$$

3.1.2 Detecting Generative Deviation via VAE

To detect nonlinear deviations not necessarily aligned with global principal directions, we complement the SVD analysis with a generative reconstruction approach using a VAE. Let $\mathbf{x}_i := (\mathbf{G}_l^{\perp})_{i,*} \in \mathbb{R}^{n_l}$ denote the i-th row of the projected residual gradient for layer l. The VAE is trained to reconstruct benign gradients via the mapping $\mathbf{x}_i \mapsto \hat{\mathbf{x}}_i$, optimized with the standard objective:

$$\mathcal{L}_{\text{vae}}(\mathbf{x}_i) = \text{MSE}(\mathbf{x}_i, \hat{\mathbf{x}}_i) + D_{\text{KL}}(q(\mathbf{z} \mid \mathbf{x}_i) \parallel p(\mathbf{z})),$$

where $\mathbf{z} \in \mathbb{R}^d$ is the latent representation and $q(\mathbf{z} \mid \mathbf{x}_i)$ is the approximate posterior. $\text{MSE}(\mathbf{x}_i, \hat{\mathbf{x}}_i) = \frac{1}{n_l} \|\mathbf{x}_i - \hat{\mathbf{x}}_i\|_2^2$ denotes the mean squared reconstruction error over the n_l dimensions, while the KL term regularizes the latent space by encouraging proximity to the prior $p(\mathbf{z})$. After training, reconstruction errors are computed as:

$$e_i := \frac{1}{n_l} \|\mathbf{x}_i - \hat{\mathbf{x}}_i\|_2^2.$$

Let μ_e, σ_e be the mean and standard deviation of $\{e_i\}$ over all rows. We flag generative outliers by

$$S_{\text{vae}} := \{i \in \{0, \ldots, m_l - 1\} \mid e_i > \mu_e + \tau \cdot \sigma_e\}.$$

This criterion identifies updates that deviate from the learned manifold of clean gradients, even when such deviations do not align with any principal direction captured by SVD.

3.1.3 Unified Detection Set

We perform T rounds of gradient accumulation and outlier detection. In each iteration t, the detected spectral and generative outliers for layer l are stored as $S_{\text{svd}}^{(l,t)}$ and $S_{\text{vae}}^{(l,t)}$, which are sequentially

accumulated into global collections (i.e., sets of per-round index sets): $\mathcal{O}^{(l)}_{\text{svd}} \leftarrow \mathcal{O}^{(l)}_{\text{svd}} \cup \{S^{(l,t)}_{\text{svd}}\}$, $\mathcal{O}^{(l)}_{\text{vae}} \leftarrow \mathcal{O}^{(l)}_{\text{vae}} \cup \{S^{(l,t)}_{\text{vae}}\}$, with global initialization $\mathcal{O}^{(l)}_{\text{svd}} \leftarrow \varnothing$ and $\mathcal{O}^{(l)}_{\text{vae}} \leftarrow \varnothing$. After T rounds, we aggregate per-round detections by $\mathcal{S}^{(l)}_{\text{svd}} \leftarrow \bigcup \mathcal{O}^{(l)}_{\text{svd}}$, $\mathcal{S}^{(l)}_{\text{vae}} \leftarrow \bigcup \mathcal{O}^{(l)}_{\text{vae}}$. The final outlier set is then defined as their union:

$$\mathcal{S}^{(l)}_{\text{union}} := \mathcal{S}^{(l)}_{\text{svd}} \cup \mathcal{S}^{(l)}_{\text{vae}}.$$

Its complement, $\mathcal{S}^{(l)}_{\text{normal}} := \{0, \ldots, m_l - 1\} \setminus \mathcal{S}^{(l)}_{\text{union}}$, identifies the structurally clean rows, which serve as the statistical reference for the correction phase.

3.2 Correction Strategy

We adopt a *prototype-based* correction that replaces each anomalous row with an empirical in-layer prototype estimated from structurally clean rows. Concretely, we compute a *single reference vector* $\bar{\mathbf{w}}^{(l)}$ as the mean of the rows in $\mathcal{S}^{(l)}_{\text{normal}}$:

$$\bar{\mathbf{w}}^{(l)} = \begin{cases} \dfrac{1}{\left|\mathcal{S}^{(l)}_{\text{normal}}\right|} \displaystyle\sum_{i \in \mathcal{S}^{(l)}_{\text{normal}}} \mathbf{W}^{(l)}_{i,*}, & \text{if } \left|\mathcal{S}^{(l)}_{\text{normal}}\right| > 0, \\ \dfrac{1}{m_l} \displaystyle\sum_{i=0}^{m_l - 1} \mathbf{W}^{(l)}_{i,*}, & \text{otherwise.} \end{cases}$$

The correction is then applied by *row-level replacement* for all union-flagged outliers:

$$\forall\, i \in \mathcal{S}^{(l)}_{\text{union}} : \qquad \mathbf{W}^{(l)}_{i,*} \leftarrow \bar{\mathbf{w}}^{(l)}.$$

The strategy is (i) *bias-reducing*, pulling anomalous rows toward the empirical centroid of clean updates; (ii) *structure-preserving*, since clean rows are unchanged and the prototype lies within the layer's observed parameter manifold; and (iii) *idempotent*, because once an outlier matches the prototype, further applications cause no drift.

3.3 Top-K Vulnerable Layer Localization

We apply *DiSec* (detection + correction) only to the *top-K vulnerable transformer layers*, since trigger-induced neuron divergence concentrates in a limited depth range rather than across all layers as discovered by our analysis (see Section 3.3.2). Because we assume no access to the true trigger, we first mine a *proxy* universal trigger from the clean auxiliary text, then use the trigger to probe how internal neuron activations shift under perturbation, and finally select the layers where the trigger induces the strongest divergence from clean behavior.

3.3.1 Adversarial Trigger Mining

We mine universal adversarial triggers using a HotFlip-inspired discrete optimization with key modifications. Classical HotFlip (Ebrahimi et al., 2018) computes token-wise gradients at selected input positions and greedily replaces a single token to maximize the loss increase under a first-order Taylor approximation. In contrast, we optimize a fixed-length **multi-token** trigger $\mathbf{t} = (t_1, \ldots, t_r)$ that is inserted into every auxiliary input x, placed immediately after `[CLS]` (i.e., at the start of the sequence). At each iteration, we compute the loss $\mathcal{L}(f(x \oplus \mathbf{t}))$, where f is the backdoored model and $\oplus$ denotes trigger insertion, and backpropagate to obtain gradients on the embedding matrix $E \in \mathbb{R}^{|V| \times d}$, where $|V|$ is the vocabulary size and d is the embedding dimension. For each trigger slot $j \in \{1, \ldots, r\}$, let g_j denote the gradient direction used to update slot j. We perform a HotFlip-style dot-product update by selecting

$$t_j \leftarrow \arg\max_{v \in V \setminus \mathcal{B}} \langle E_v,\, g_j \rangle\,,$$

where $\mathcal{B}$ is a banned set of special tokens, including `[CLS]`, `[SEP]`, `[PAD]`, and `[UNK]`. Repeating this greedy replacement across trigger slots and iterations yields **multi-token universal adversarial sequences** that consistently induce high-loss behavior across diverse contexts.

Importantly, the objective is **label-free**: we define $\mathcal{L}$ as cross-entropy with respect to the model's own predicted label (pseudo-label) and greedily *increase* this self-loss via gradient-based token replacement using only unlabeled auxiliary text, making the trigger mining independent of task-specific supervision or any trigger/target knowledge.

3.3.2 Selective Neuron Activation Analysis

Prior work shows that PLMs exhibit selective activation, where different tasks stimulate different neuron subsets (Wang et al., 2022). Extending this, Zhu et al. (2023) find that poisoned inputs in backdoored models selectively activate a distinct neuron set, implying a structural footprint of adversarial contamination. Motivated by these findings, we design a trigger-agnostic overlap-ratio analysis to compare activation patterns of a backdoored model under clean versus adversarially perturbed inputs.

We evaluate the backdoored model on a held-out set of n clean sentences sampled from the auxiliary corpus, running two forward passes per example: (i) the original input and (ii) the same input with the mined trigger from Section 3.3.1 injected (e.g., immediately after `[CLS]`). From each pass, we extract the `[CLS]` hidden states from all transformer

layers. For each layer ℓ and example i, we binarize the [CLS] vector by marking a neuron active if its scalar value is positive; let $A_{\ell,i}$ be the number of neurons active in both the clean and triggered runs, and $B_{\ell,i}$ the number active under the triggered run. We define the layer-wise *overlap ratio* as $r_\ell = \sum_i A_{\ell,i} / \sum_i B_{\ell,i}$, i.e., the ratio of intersection to triggered activations, pooled over the entire evaluation set. A low overlap ratio indicates that the trigger activates neurons that are largely dormant on clean data, revealing distinct adversarial pathways induced by the backdoor and a higher ratio suggests that the trigger primarily reuses neurons already active for benign inputs.

3.3.3 Top-K Layer Selection

To localize where backdoor effects emerge, we examine how overlap changes across adjacent layers via

$$\Delta r_\ell \;=\; r_\ell - r_{\ell-1}, \qquad \ell \geq 2.$$

A large negative Δr_ℓ indicates an abrupt shift toward trigger-activated neurons that are largely inactive on clean inputs. Our selection procedure has three steps: *(1) spike-aware onset via sharpest drop*, which sets the onset at the most negative Δr_ℓ while *excluding* drops that immediately follow rare rebound spikes; *(2) backward expansion* to optionally include a preceding substantial drop; and *(3) forward continuation with rebound-spike filtering* to remove rare layers with unusually large positive rebounds. The resulting region defines the top-K layers $\mathcal{L}_{\text{top-}K}$ targeted by *DiSec*. Full selection details are in the Appendix (Section H), and the complete *DiSec* pipeline is summarized in Appendix Algorithm 2.

4 Experiments

Experimental Setting. We evaluate *DiSec* against three task-agnostic backdoor attacks injected into $\text{BERT}_{\text{BASE-UNCASED}}$—BadPre (Chen et al., 2022a), POR (Shen et al., 2021), and NeuBA (Zhang et al., 2023)—and additionally on NeuBA-poisoned $\text{RoBERTA}_{\text{BASE}}$. As clean reference checkpoints, we use the original `bert-base-uncased` and `roberta-base` models from the HuggingFace hub, enabling controlled comparisons on identical transformer architectures with and without injected backdoors. All experiments are run on a single NVIDIA A100 GPU (40 GB). Following Zhu et al. (2023), we use 20k randomly sampled BookCorpus sentences (Zhu et al., 2015) as unlabeled auxiliary text. Unlike their MLM-based objective, our gradient accumulation uses a classification-style loss. We therefore assign a label-efficient uniform pseudo-label (i.e., no human labels) to instantiate the loss, and apply a minimal length filter (≥ 5 tokens) to remove trivially short inputs. This lets *DiSec* estimate layer-wise outlier statistics without any labeled downstream data.

For downstream evaluation, we fine-tune purified models on SST-2 sentiment analysis (Socher et al., 2013), Hate Speech and Offensive Language (HSOL) toxicity detection (Davidson et al., 2017), and AG News topic classification (Zhang et al., 2015). Following Chen et al. (2022d), we consolidate HSOL labels by merging both hate and offensive into a single `hate` class and treating all remaining instances as `non-hate`. Detailed dataset statistics and HSOL preprocessing are provided in the Appendix (Sections B and E).

Evaluation Metrics. We evaluate models using Clean Accuracy (ACC), Average Attack Success Rate (AASR), and Maximum Attack Success Rate (MASR). Details on metric definitions and poisoned test set construction are provided in Appendix Section D.

Baselines. To contextualize *DiSec*'s effectiveness, we compare against label-free defenses spanning input filtering and model purification under the same constraints (no downstream labels; only 20k BookCorpus sentences for statistics collection). *w/o defense* fine-tunes the backdoored model on each downstream task without any countermeasure.

Fine-Pruning (FP) (Liu et al., 2018) removes neurons in the *last* transformer block with the smallest average activation on the auxiliary corpus; following the original protocol, we collect [CLS] activations, rank neurons, and prune the bottom 20%. Global Fine-Pruning (FP-G) generalises FP to all twelve encoder layers, yielding two variants. FP-GM prunes 20% of neurons layer-wise, whereas FP-GA doubles the ratio to 40%.

ONION (Qi et al., 2021a) is an inference-time input filter that deletes tokens whose removal reduces GPT-2 perplexity beyond a threshold, and then feeds the sanitized sentence to the unmodified backbone. We evaluate it under an optimistic, upper-bound configuration intended to benefit its detection capability by setting the threshold to 0.

RECIPE (Zhu et al., 2023) demonstrates weight-space purification through auxiliary masked-language-modeling (MLM) with activation spar-

sity regularization. Following the original authors' guidance, we perform masked-language-model pretraining on our BookCorpus subset for 4, 8, and 10 epochs for the POR, BadPre, and NeuBA backdoor-injected models, respectively. During this phase, we freeze all model parameters except the `intermediate.dense.weight` layers and optimize using AdamW with a learning rate of 2×10^{-5}. The purified weights replace the original backbone before downstream fine-tuning.

Defense Settings. *DiSec* operates in a strictly label-free setting using only 20k BookCorpus sentences; no downstream labels, adversarial triggers, or poisoned inputs are exposed during purification.

We cache the original weights and operate on the top-K attention layers of the encoder. For each target weight matrix, we accumulate gradients on the auxiliary corpus and subtract the component collinear with the stored weights, producing a per-row residual that highlights anomalous directions. Residuals are screened by two detectors. *SVD*: we compute a thin SVD, retain the top k components, scale left singular vectors by their singular values to obtain per-row coefficients, convert absolute coefficients to z-scores, and flag a row when its maximum z exceeds the layer mean by more than $\tau = 2$ standard deviations. *VAE*: we train a lightweight auto-encoder (128–64–16 latent; ReLU; Adam with lr $= 10^{-3}$; 10 epochs; batch size 32) on the same residual and rows whose reconstruction error is more than $\tau = 2$ standard deviations above the mean are flagged.

Detection runs for T iterations, pooling new flagged rows each pass to form the suspect set. We set the SVD rank to k=30 and use $T \in \{1, 1, 3, 1\}$ detection iterations for POR, BadPre, NeuBA-BERT, and NeuBA-RoBERTa, respectively. These checkpoint-specific values are selected via a task-agnostic Jaccard-based stability criterion without using any downstream data. Details are provided in Appendix Section J.

Full Fine-Tuning Settings. We use an identical full fine-tuning protocol for all defense variants (including *DiSec* and all re-implemented baselines) to isolate defense impact from optimization artifacts. All models are fine-tuned end-to-end for 2 epochs with batch size 32 using AdamW over all parameters (lr $= 2\times10^{-5}$) and a linear learning-rate scheduler without warmup, with total steps set to $|\texttt{train}| \times \texttt{epochs}$ (where $|\texttt{train}|$ denotes the number of training examples) for smooth decay throughout training. Unless stated otherwise, we report a single run per configuration with a fixed seed. We keep the same seed and setup across all defenses, baselines, and ablations.

Settings for Top-K Layer Localization. We mine a fixed-length universal trigger $\mathbf{t} = (t_1, \ldots, t_r)$ with r=5 tokens and insert it after `[CLS]`. See Appendix Section G for trigger-mining details. For overlap ratios, we sample a held-out set of n=200 sentences from the 20K auxiliary corpus and run paired clean vs. trigger-injected forward passes to compute layer-wise activation overlap.

5 Result Analysis

5.1 Results of Top-K Layer Localization

Figure 1 illustrates how trigger-induced activation patterns evolve across layers and motivates our top-K selection. The top row plots Δr_ℓ across layers, where pronounced negative drops indicate abrupt transitions into trigger-dominated computation. Applying our selection procedure (Appendix, Section H), we obtain the following top-K vulnerable attention layers (1-based indexing): POR-BERT: 3∼12 (K=10), BadPre-BERT: 9∼12 (K=4), NeuBA-BERT: 2∼12 (K=11), and NeuBA-RoBERTa: $\{2\sim10, 12\}$ (K=10), excluding layer 11 due to a rebound spike.

To validate these trigger-agnostic selections, we repeat the overlap analysis using the *actual* adversarial triggers for each backdoored model (bottom row of Figure 1). We compute overlap ratios for clean, backdoored models and mark the first layer where clean and backdoored curves diverge substantially (vertical dashed line). These divergence points match the top-K layers found using only the mined trigger, showing we can localize backdoor-sensitive layers without the true trigger.

5.2 Purification of Backdoored Models

We apply *DiSec* only to the selected top-K vulnerable layers for each backdoored model and compare it with existing defenses in Table 1. Across all four backdoored PLMs and three datasets, *DiSec* consistently reduces AASR and MASR to very low levels while maintaining competitive ACC, whereas some baselines perform well on a single dataset but do not generalize. For POR BERT / AG News, ONION achieves a slightly lower MASR than *DiSec* (3.84% vs. 4.04%). However, ONION degrades on other datasets, e.g., NeuBA BERT / HSOL where its AASR and MASR remain high (48.78%, 87.77%) compared to *DiSec* (5.13%,

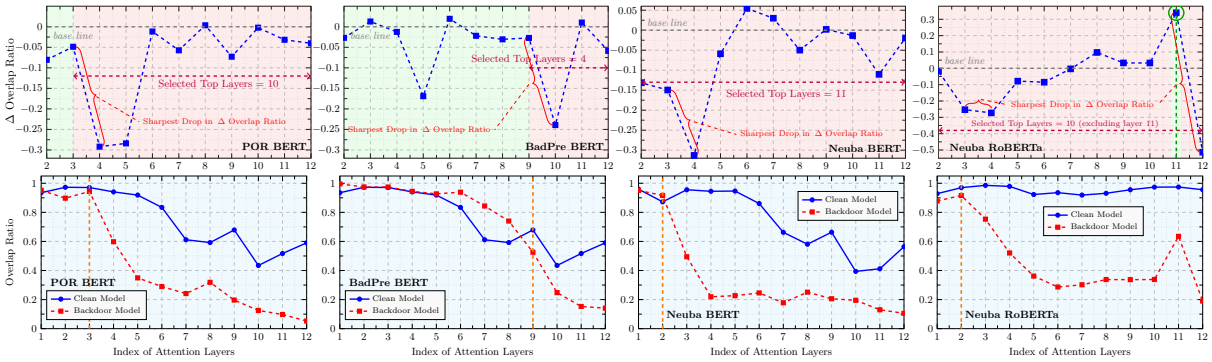

Figure 1: ***Top:*** Change of Neuron activation overlap ratios in different backdoor induced models using triggers mined from Section 3.3.1. ***Bottom:*** Neuron activation overlap ratios for clean, and backdoored models assuming access to the true adversarial triggers.

5.52%). Similarly, RECIPE yields the best AASR on BadPre BERT / SST-2 (10.28% vs. 10.30% for *DiSec*) and performs well on NeuBA BERT / SST-2 (11.46% vs. 11.79%), but this robustness does not transfer: on NeuBA BERT, RECIPE's (AASR, MASR) jumps to (16.09%, 60.69%) on HSOL and (18.46%, 48.01%) on AG News, whereas *DiSec* keeps them low at (5.13%, 5.52%) and (3.23%, 3.75%), respectively. Overall, even in the few cases where *DiSec* is not strictly optimal on a single metric, its AASR and MASR remain within a small margin of the best results, yielding a defense that is robust and stable across datasets and backdoor constructions.

5.3 Purification of Clean Models

Table 2 studies whether *DiSec* harms *clean* pretrained models when applied under the same top-K selection protocol used for backdoored checkpoints (Appendix H). For clean BERT and RoBERTa, our selection procedure identifies $K{=}4$ and $K{=}2$ vulnerable layers, respectively, even though these models contain no backdoor. Applying *DiSec* to these selected layers leaves downstream accuracy essentially unchanged: on SST-2, accuracy slightly improves (BERT: 90.94→91.10; RoBERTa: 94.34→94.67), and for BERT it also increases on AG News by $+0.21$ (94.42→94.63). We attribute these small gains to the mild regularization effect of our row-level correction, while the remaining changes are negligible (e.g., HSOL drops by only 0.24 for BERT and 0.49 for RoBERTa).

To further stress-test stability, we also report results when applying *DiSec* to a larger set of layers (Top-9) for both models. Even under this more aggressive setting, the accuracy changes remain modest (within -0.77 for BERT and -2.03 for RoBERTa on SST-2, and within about one point on HSOL/AG News), indicating that *DiSec* does not introduce substantial degradation when used on clean checkpoints. Overall, these results confirm that our defense is conservative on benign models while remaining effective for genuinely backdoored ones.

Models	Datasets	SST-2			HSOL			AG News		
	Methods	ACC	AASR	MASR	ACC	AASR	MASR	ACC	AASR	MASR
POR BERT	w/o defense	91.60	99.54	100.00	96.29	46.63	99.52	94.43	16.26	32.51
	RECIPE	89.51	39.60	82.95	95.47	29.28	72.73	94.31	4.78	6.04
	ONION	91.60	17.09	19.63	95.89	39.11	79.14	94.50	3.79	**3.84**
	FP	91.76	99.65	100.00	95.80	50.35	98.80	94.66	14.05	27.12
	FP-GM	90.83	98.77	100.00	95.93	15.73	55.40	94.47	24.96	62.77
	FP-GA	90.88	82.98	100.00	95.80	6.05	8.63	94.53	11.93	28.39
	DiSec	89.02	**13.12**	**15.68**	95.56	**5.61**	**6.24**	94.57	**3.42**	4.04
BadPre BERT	w/o defense	91.27	98.13	98.68	95.93	84.46	90.89	94.34	48.49	53.54
	RECIPE	90.17	**10.28**	10.78	96.46	20.39	34.15	94.48	9.77	13.85
	ONION	91.27	16.40	17.00	95.97	28.07	30.99	94.49	3.98	4.09
	FP	91.43	97.80	98.24	96.01	6.33	7.23	94.53	43.67	47.11
	FP-GM	90.66	28.98	31.79	95.85	4.65	4.80	94.41	16.36	18.89
	FP-GA	90.12	11.75	12.76	95.56	6.38	7.19	94.53	3.34	3.77
	DiSec	90.88	10.30	**10.67**	96.13	**4.36**	**4.41**	94.43	**3.05**	**3.09**
NeuBA BERT	w/o defense	91.27	99.95	100.00	96.21	43.76	100.00	94.53	82.39	99.61
	RECIPE	89.62	**11.46**	**12.43**	96.17	16.09	60.69	94.62	18.46	48.01
	ONION	91.27	22.93	34.98	96.01	48.78	87.77	94.53	4.27	5.18
	FP	91.16	99.91	100.00	96.09	51.92	99.76	94.47	77.45	99.19
	FP-GM	90.83	57.85	100.00	96.09	25.29	98.32	94.50	28.22	61.21
	FP-GA	89.62	19.34	44.85	95.60	9.38	29.98	94.38	3.97	6.02
	DiSec	88.85	11.79	13.93	95.68	**5.13**	**5.52**	94.39	**3.23**	**3.75**
NeuBA RoBERTa	w/o defense	94.67	75.80	93.97	96.37	72.43	91.61	95.01	72.99	97.53
	RECIPE	92.81	27.76	58.09	95.97	52.78	86.73	94.83	54.19	96.97
	ONION	94.67	12.79	**13.60**	95.64	20.96	28.86	94.87	3.28	**3.42**
	FP	94.23	72.91	93.97	95.68	59.24	89.69	95.80	47.82	89.45
	FP-GM	93.52	57.45	93.97	95.80	47.82	89.45	94.92	68.95	97.42
	FP-GA	91.82	52.08	93.53	95.56	35.68	87.29	94.70	39.18	88.12
	DiSec	88.96	**12.20**	14.69	95.40	**6.69**	**8.63**	94.50	**3.21**	4.12

Table 1: ACC%, AASR%, and MASR% of our *DiSec* defense and existing baselines evaluated against four backdoored models for comparison.

5.4 Robustness Against Adaptive Attacks

We further evaluate *DiSec* under an adaptive threat model in which the attacker is aware of the defender's purification strategy and modifies the poisoning objective accordingly. Building on BadPre, we construct a defense-aware variant that introduces an anomaly regularizer to suppress the residual-gradient signal exploited by *DiSec*. The attacker optimizes

$$\mathcal{L}_{\text{adv}} = \mathcal{L}_{\text{clean}} + \mathcal{L}_{\text{poison}} + \mathcal{L}_{\text{anom}}$$

Clean Models	Overlap Ratios (Layer 1 ∼ Layer 12)	Datasets	SST-2	HSOL	AG News
		Methods	ACC	ACC	ACC
BERT	0.9375, 0.8686, 0.9123, **0.9324**,	*w/o defense*	90.94$_{(0.00)}$	96.17$_{(0.00)}$	94.42$_{(0.00)}$
	0.7969, 0.7175, 0.6376, 0.5641,	DiSec (Top 9)	90.17$_{(-0.77)}$	95.64$_{(-0.53)}$	94.76$_{(+0.34)}$
	0.5539, **0.4169**, 0.5137, 0.5523	**DiSec* (Top 4)**	91.10$_{(+0.16)}$	95.93$_{(-0.24)}$	94.63$_{(+0.21)}$
RoBERTa	0.8536, 0.9273, 0.9437, **0.9810**,	*w/o defense*	94.34$_{(0.00)}$	96.29$_{(0.00)}$	94.89$_{(0.00)}$
	0.9284, 0.9508, 0.9465, 0.9352,	DiSec (Top 9)	92.31$_{(-2.03)}$	95.36$_{(-0.93)}$	94.66$_{(-0.23)}$
	0.9490, 0.9532, **0.9852**, **0.8374**	**DiSec* (Top 2)**	94.67$_{(+0.33)}$	95.80$_{(-0.49)}$	94.86$_{(-0.03)}$

Table 2: ACC% of clean BERT and RoBERTa models after applying *DiSec*. Subscripts show relative change from the corresponding *w/o defense* baseline for each dataset. Red entries in the overlap-ratio column mark layers around the sharpest drops.

Datasets	SST-2			HSOL			AG News		
Methods	**ACC**	**AASR**	**MASR**	**ACC**	**AASR**	**MASR**	**ACC**	**AASR**	**MASR**
w/o defense	90.28	100.00	100.00	95.72	100.00	100.00	94.33	32.26	90.74
DiSec	90.50	19.45	23.76	95.76	5.70	8.34	94.46	2.99	3.16

Table 3: Performance of *DiSec* under the adaptive BadPre attack on SST-2, HSOL, and AG News.

Methods	RECIPE	ONION	FP	FP-GM	FP-GA	**DiSec**
Time (s)	189.93	1451.40	97.91	226.35	238.60	**174.79**

Table 4: Runtime in seconds for different defense pipelines on SST-2 against POR attack.

Models	Datasets	SST-2			HSOL			AG News		
	Methods	**ACC**	**AASR**	**MASR**	**ACC**	**AASR**	**MASR**	**ACC**	**AASR**	**MASR**
POR BERT	VAE-*only*	89.62	15.79	22.99	95.68	6.47	7.43	94.39	4.26	5.37
	SVD-*only*	89.68	**12.25**	**13.42**	95.64	6.24	7.19	94.42	3.75	5.07
	VAE ∩ SVD	90.23	82.90	100.00	95.52	20.00	44.36	94.43	3.98	4.77
	VAE ∪ SVD	89.02	13.12	15.68	95.56	**5.61**	**6.24**	94.57	**3.42**	**4.04**
BadPre BERT	VAE-*only*	91.49	39.50	41.45	96.05	6.35	7.47	94.55	6.15	6.60
	SVD-*only*	90.99	21.36	21.93	95.97	4.96	5.14	94.55	5.99	6.21
	VAE ∩ SVD	90.83	66.67	68.43	96.01	5.44	5.92	94.66	8.49	8.88
	VAE ∪ SVD	90.88	**10.30**	**10.67**	96.13	**4.36**	**4.41**	94.43	**3.05**	**3.09**
NeuBA BERT	VAE-*only*	90.88	21.33	33.11	95.89	10.29	23.02	94.42	3.49	4.54
	SVD-*only*	89.35	14.35	22.15	95.56	**5.04**	**5.52**	94.46	3.27	3.82
	VAE ∩ SVD	90.33	19.65	37.50	95.40	19.15	36.45	94.49	3.42	4.60
	VAE ∪ SVD	88.85	**11.79**	**13.93**	95.68	5.13	**5.52**	94.39	**3.23**	**3.75**
NeuBA RoBERTa	VAE-*only*	90.55	28.82	79.06	95.60	21.82	59.71	94.74	13.87	68.05
	SVD-*only*	88.03	32.44	83.88	95.40	12.04	40.53	94.41	8.53	31.56
	VAE ∩ SVD	90.39	66.65	94.28	95.60	47.43	89.45	94.78	28.52	87.42
	VAE ∪ SVD	88.96	**12.20**	**14.69**	95.40	**6.69**	**8.63**	94.50	**3.21**	**4.12**

Table 5: Ablation Study: ACC%, AASR%, and MASR% of different defense variants with prototype-based mean correction across four backdoor models.

where $\mathcal{L}_{\text{clean}}$ is the standard masked language modeling loss on clean auxiliary data, $\mathcal{L}_{\text{poison}}$ is the poisoned MLM loss on trigger-inserted samples with random labels at masked positions, and $\mathcal{L}_{\text{anom}}$ is the average residual-gradient energy over the query, key, and value matrices in the Top-K vulnerable layers. Minimizing $\mathcal{L}_{\text{anom}}$ encourages the attacker to make the poisoned model less distinguishable under *DiSec*'s detection criterion. Nevertheless, Table 3 shows that *DiSec* remains effective in this stronger setting, reducing AASR on SST-2, HSOL, and AG News from 100.00%, 100.00%, and 32.26% to 19.45%, 5.70%, and 2.99%, respectively.

5.5 Runtime Analysis

We evaluate the runtime of different defense pipelines on SST-2 against the POR-BERT backdoor attack. All methods, including purification and subsequent fine-tuning, are run on a single NVIDIA A100 GPU with 40GB memory. As shown in Table 4, *DiSec* requires 174.79 seconds, reflecting a moderate computational cost. ONION is the slowest method by a large margin, requiring 1451.40 seconds, whereas FP is the fastest at 97.91 seconds. RECIPE, FP-GM, and FP-GA require 189.93, 226.35, and 238.60 seconds, respectively. These results suggest that *DiSec* offers a favorable efficiency-effectiveness trade-off and is considerably more practical than ONION.

5.6 Ablation Study

Table 5 shows that both components of *DiSec* are necessary. *DiSec* uses the union detector (**VAE ∪ SVD**) with prototype-based mean correction. VAE-*only* and SVD-*only* can reduce AASR and MASR compared to the backdoored baselines, but their improvements are highly model- and dataset-dependent: for example, SVD-*only* performs best on POR BERT / SST-2 (12.25% AASR, 13.42% MASR), whereas on BadPre BERT / SST-2 it remains much higher (21.36%, 21.93%) than the best achievable values in that setting. Intersection variant (VAE ∩ SVD) is overly conservative, misses many outliers, and leaves high AASR, MASR in several settings. Overall, **VAE ∪ SVD** achieves the best or near-best trade-off in most cases while keeping ACC close to the original models.

Table 6 shows the performance of three alternative Row-Correction (RC) strategies: **RC1** replaces each outlier row with the corresponding row from a clean pretrained model, **RC2** adds Gaussian noise scaled by the row's own standard deviation (local noise), and **RC3** adds Gaussian noise scaled by a single global standard deviation for the layer's weight matrix (global noise). Compared with our prototype-based mean correction, RC3 performs better on POR BERT / HSOL (5.47%, 6.21%) and NeuBA RoBERTa / HSOL (5.79%, 7.61%) (i.e., the four bold numbers in Table 6), but our mean correction method performs best in all other cases. Overall, these results suggest that simple noise injection or copying rows from a clean model is not reliably effective, whereas prototype-based correction is key to achieving consistent robustness.

Finally, Table 7 examines a "blind" variant that applies *DiSec* to *all 12 layers* rather than the selected top-K vulnerable layers. This set-

Models	Datasets	SST-2			HSOL			AG News		
	Methods	ACC	AASR	MASR	ACC	AASR	MASR	ACC	AASR	MASR
POR BERT	DiSec (RC1)	91.21	99.45	100.00	95.76	26.10	46.27	94.39	22.87	50.91
	DiSec (RC2)	89.68	38.91	62.17	95.80	5.80	6.24	94.21	3.54	4.25
	DiSec (RC3)	90.23	37.62	59.98	95.48	**5.47**	**6.21**	94.36	3.74	4.35
BadPre BERT	DiSec (RC1)	91.87	84.14	85.81	95.89	6.10	7.57	94.58	3.75	3.88
	DiSec (RC2)	90.99	98.15	98.46	95.93	5.34	6.11	94.38	8.76	10.35
	DiSec (RC3)	91.21	96.44	96.92	96.01	5.69	6.79	94.46	17.06	18.44
NeuBA BERT	DiSec (RC1)	91.98	53.81	100.00	95.97	29.02	96.99	94.57	4.77	13.12
	DiSec (RC2)	89.79	22.76	59.76	95.76	5.45	6.24	94.30	3.63	4.37
	DiSec (RC3)	87.92	18.62	30.70	95.80	5.20	6.00	94.32	3.59	4.70
NeuBA RoBERTa	DiSec (RC1)	95.00	54.09	94.17	96.05	34.81	89.21	94.70	32.53	85.70
	DiSec (RC2)	79.96	29.69	46.97	95.36	17.12	71.70	93.83	11.35	48.95
	DiSec (RC3)	77.65	48.54	82.18	95.04	**5.79**	**7.61**	93.76	5.98	19.28

Table 6: Ablation study on alternate correction strategies for the *DiSec* defense. The four bold numbers are the only cases where the variants outperform our main *DiSec* method.

Backdoored Models	SST-2			HSOL			AG News		
	ACC	AASR	MASR	ACC	AASR	MASR	ACC	AASR	MASR
POR	88.30$_{(-0.72)}$	**12.52**$_{(-0.60)}$	**13.49**$_{(-2.19)}$	**95.76**$_{(+0.20)}$	**5.47**$_{(-0.14)}$	**6.00**$_{(-0.24)}$	94.41$_{(-0.16)}$	**3.29**$_{(-0.13)}$	**3.65**$_{(-0.39)}$
BadPre	87.81$_{(-3.07)}$	12.88$_{(+2.58)}$	13.49$_{(+2.82)}$	95.44$_{(-0.69)}$	5.52$_{(+1.16)}$	5.76$_{(+1.35)}$	94.18$_{(-0.25)}$	3.26$_{(+0.21)}$	3.30$_{(+0.21)}$
N. BERT	88.63$_{(-0.22)}$	13.00$_{(+1.21)}$	14.25$_{(+0.32)}$	95.64$_{(-0.04)}$	5.17$_{(+0.04)}$	6.95$_{(+1.43)}$	94.05$_{(-0.34)}$	3.37$_{(+0.14)}$	3.86$_{(+0.11)}$
N. RoBERTa	87.10$_{(-1.86)}$	15.55$_{(+3.35)}$	18.42$_{(+3.73)}$	95.40$_{(0.00)}$	**5.75**$_{(-0.94)}$	**6.47**$_{(-2.16)}$	94.24$_{(-0.26)}$	3.22$_{(+0.01)}$	4.63$_{(+0.51)}$

Table 7: Performance of our defense when blindly applying to *all 12 layers* of each backdoored model. Each entry shows the value with a subscript indicating the change relative to *DiSec* (Selective top-K) on the same model and dataset. Here, N. represents Neuba induced backdoor models.

ting is only reliably effective for the POR checkpoint, where blind all-layer purification still yields strong robustness across datasets. However, this behavior does not generalize: for BadPre and both NeuBA checkpoints, it typically reduces ACC (e.g., BadPre/SST-2: -3.07) and increases (AASR, MASR) relative to selective top-K purification (e.g., BadPre/SST-2: $+2.58, +2.82$; NeuBA RoBERTa/SST-2: $+3.35, +3.73$). Overall, these ablations show that (i) the **VAE** $\cup$ **SVD** union rule, (ii) prototype-based mean correction, and (iii) selective top-K targeting are all necessary for robust and stable backdoor removal. See Appendix Section K for additional ablations on top-K targeting and SVD/VAE detector contributions.

Table 8 examines the effect of auxiliary corpus size on *DiSec* against the POR attack across three downstream tasks. Varying the auxiliary set size around the default 20k samples does not lead to a consistent degradation in downstream performance. Notably, smaller auxiliary sets often achieve results comparable to, and in some cases slightly better than, those obtained with 20k samples. This suggests that *DiSec* is relatively insensitive to the exact size of the clean auxiliary corpus, which is desirable for practical deployment. Table 9 studies whether *DiSec* remains effective when the auxiliary corpus is replaced. Here, in place of BookCorpus (Zhu et al., 2015), we use an auxiliary corpus formed by randomly sampling and cleaning 20k sentence instances from Hugging Face's `Skylion007/openwebtext` (Gokaslan et al., 2019). The results show that *DiSec* continues to achieve strong performance across POR, BadPre, and NeuBA attacks on all downstream datasets. This demonstrates that the proposed method is not tied to a particular auxiliary corpus and can operate effectively with different sources of clean text.

Datasets	SST-2			HSOL			AG News		
Size	ACC	AASR	MASR	ACC	AASR	MASR	ACC	AASR	MASR
1k	89.73	11.05	12.54	95.68	5.80	7.43	94.45	3.41	3.86
5k	89.57	12.30	13.75	95.56	5.47	6.24	94.63	3.53	3.98
10k	88.74	13.36	15.35	95.72	5.71	6.26	94.51	3.61	4.05
15k	89.29	12.66	14.14	95.80	5.13	6.00	94.61	3.50	4.12
20k	**89.02**	**13.12**	**15.68**	**95.56**	**5.61**	**6.24**	**94.57**	**3.42**	**4.04**
25k	89.24	11.76	13.64	95.52	5.32	6.47	94.37	3.53	4.05

Table 8: Effect of auxiliary dataset size on *DiSec* against the POR attack across three downstream datasets.

Datasets	SST-2			HSOL			AG News		
Models	ACC	AASR	MASR	ACC	AASR	MASR	ACC	AASR	MASR
POR-BERT	89.57	11.88	13.05	95.68	5.33	6.71	94.53	3.28	3.74
BadPre-BERT	91.05	11.16	11.33	95.76	4.46	4.56	94.63	3.25	3.28
NeuBA-BERT	89.18	12.70	14.14	95.60	5.15	5.76	94.33	3.29	3.95

Table 9: Results of *DiSec* using different auxiliary data against three different backdoor attacks.

6 Conclusion

We presented *DiSec*, a trigger-agnostic defense for backdoored PLMs that combines dual outlier detection (SVD + VAE), prototype-based row correction, and a data-driven top-K layer selection based on neuron-activation overlap ratio. Without accessing the true trigger or task labels, *DiSec* substantially reduces AASR, MASR on four backdoored BERT/RoBERTa models across SST-2, HSOL, and AG News, while largely preserving clean accuracy. Ablation studies show that both detectors, prototype-based correction, and selective top-K purification are necessary, and that blindly correcting all layers or using naive noise-based corrections is significantly less effective. Although *DiSec* is heuristic and assumes white-box access, it offers a practical and model-centric step toward structurally purifying backdoored language models.

Limitations

DiSec assumes white-box access to model parameters, which may not hold in fully proprietary or API-only settings. Although selective top-K purification keeps clean accuracy largely intact, it still modifies model weights and offers no formal robustness guarantees. Thus, *DiSec* should be regarded as a strong empirical defense rather than a provably secure one. Moreover, while the core components of *DiSec* are model-agnostic in principle,

the current study is limited to pre-trained language models for classification tasks, and extending and validating the method on backdoor-induced LLMs and more diverse task settings remains important future work.

Potential risks. *DiSec* is not a certification of model safety, and residual backdoors or adaptive attacks may remain. Because purification alters model parameters, it may remove benign capabilities or disproportionately affect specialized or under-represented inputs depending on the auxiliary data distribution. We recommend using *DiSec* as a complementary tool alongside standard security practices, including monitoring, red-teaming, and provenance tracking.

Acknowledgement

This research is supported by the Arkansas High Performance Computing Center which is funded through multiple National Science Foundation grants and the Arkansas Economic Development Commission.

References

Ansh Arora, Xuanli He, Maximilian Mozes, Srinibas Swain, Mark Dras, and Qiongkai Xu. 2024. Here'sa free lunch: Sanitizing backdoored models with model merge. In *Findings of the Association for Computational Linguistics ACL 2024*, pages 15059–15075.

Xiangrui Cai, Haidong Xu, Sihan Xu, Ying Zhang, and 1 others. 2022. Badprompt: Backdoor attacks on continuous prompts. *Advances in Neural Information Processing Systems*, 35:37068–37080.

Shuwen Chai and Jinghui Chen. 2022. One-shot neural backdoor erasing via adversarial weight masking. *Advances in Neural Information Processing Systems*, 35:22285–22299.

Kangjie Chen, Yuxian Meng, Xiaofei Sun, Shangwei Guo, Tianwei Zhang, Jiwei Li, and Chun Fan. 2022a. Badpre: Task-agnostic backdoor attacks to pre-trained NLP foundation models. In *International Conference on Learning Representations*.

Sishuo Chen, Wenkai Yang, Zhiyuan Zhang, Xiaohan Bi, and Xu Sun. 2022b. Expose backdoors on the way: A feature-based efficient defense against textual backdoor attacks. In *Findings of the Association for Computational Linguistics: EMNLP 2022*, pages 668–683.

Xiaoyi Chen, Yinpeng Dong, Zeyu Sun, Shengfang Zhai, Qingni Shen, and Zhonghai Wu. 2022c. Kallima: A clean-label framework for textual backdoor attacks. In *European Symposium on Research in Computer Security*, pages 447–466. Springer.

Yangyi Chen, Hongcheng Gao, Ganqu Cui, Fanchao Qi, Longtao Huang, Zhiyuan Liu, and Maosong Sun. 2022d. Why should adversarial perturbations be imperceptible? rethink the research paradigm in adversarial nlp. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 11222–11237.

Thomas Davidson, Dana Warmsley, Michael Macy, and Ingmar Weber. 2017. Automated hate speech detection and the problem of offensive language. In *Proceedings of the international AAAI conference on web and social media*, volume 11, pages 512–515.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 conference of the North American chapter of the association for computational linguistics: human language technologies, volume 1 (long and short papers)*, pages 4171–4186.

Wei Du, Peixuan Li, Haodong Zhao, Tianjie Ju, Ge Ren, and Gongshen Liu. 2024. Uor: Universal backdoor attacks on pre-trained language models. In *Findings of the Association for Computational Linguistics ACL 2024*, pages 7865–7877.

Javid Ebrahimi, Anyi Rao, Daniel Lowd, and Dejing Dou. 2018. Hotflip: White-box adversarial examples for text classification. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 31–36.

Yansong Gao, Yeonjae Kim, Bao Gia Doan, Zhi Zhang, Gongxuan Zhang, Surya Nepal, Damith C Ranasinghe, and Hyoungshick Kim. 2021. Design and evaluation of a multi-domain trojan detection method on deep neural networks. *IEEE Transactions on Dependable and Secure Computing*, 19(4):2349–2364.

Aaron Gokaslan, Vanya Cohen, Ellie Pavlick, and Stefanie Tellex. 2019. Openwebtext corpus. `http://Skylion007.github.io/OpenWebTextCorpus`.

Peihai Jiang, Xixiang Lyu, Yige Li, and Jing Ma. 2025. Backdoor token unlearning: Exposing and defending backdoors in pretrained language models. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pages 24285–24293.

Lesheng Jin, Zihan Wang, and Jingbo Shang. 2022. Wedef: Weakly supervised backdoor defense for text classification. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 11614–11626.

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

Yige Li, Xixiang Lyu, Nodens Koren, Lingjuan Lyu, Bo Li, and Xingjun Ma. 2021. Neural attention distillation: Erasing backdoor triggers from deep neural

networks. In *International Conference on Learning Representations*.

Kang Liu, Brendan Dolan-Gavitt, and Siddharth Garg. 2018. Fine-pruning: Defending against backdooring attacks on deep neural networks. In *International symposium on research in attacks, intrusions, and defenses*, pages 273–294. Springer.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Fanchao Qi, Yangyi Chen, Mukai Li, Yuan Yao, Zhiyuan Liu, and Maosong Sun. 2021a. Onion: A simple and effective defense against textual backdoor attacks. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 9558–9566.

Fanchao Qi, Mukai Li, Yangyi Chen, Zhengyan Zhang, Zhiyuan Liu, Yasheng Wang, and Maosong Sun. 2021b. Hidden killer: Invisible textual backdoor attacks with syntactic trigger. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 443–453.

Fanchao Qi, Yuan Yao, Sophia Xu, Zhiyuan Liu, and Maosong Sun. 2021c. Turn the combination lock: Learnable textual backdoor attacks via word substitution. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 4873–4883.

Lujia Shen, Shouling Ji, Xuhong Zhang, Jinfeng Li, Jing Chen, Jie Shi, Chengfang Fang, Jianwei Yin, and Ting Wang. 2021. Backdoor pre-trained models can transfer to all. In *Proceedings of the 2021 ACM SIGSAC Conference on Computer and Communications Security*, pages 3141–3158.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D Manning, Andrew Y Ng, and Christopher Potts. 2013. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 conference on empirical methods in natural language processing*, pages 1631–1642.

Yao Tong, Weijun Li, Xuanli He, Haolan Zhan, and Qiongkai Xu. 2025. Cut the deadwood out: Backdoor purification via guided module substitution. In *Findings of the Association for Computational Linguistics: EMNLP 2025*, pages 23760–23783.

Xiaozhi Wang, Kaiyue Wen, Zhengyan Zhang, Lei Hou, Zhiyuan Liu, and Juanzi Li. 2022. Finding skill neurons in pre-trained transformer-based language models. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 11132–11152.

Dongxian Wu and Yisen Wang. 2021. Adversarial neuron pruning purifies backdoored deep models. *Advances in Neural Information Processing Systems*, 34:16913–16925.

Wenkai Yang, Yankai Lin, Peng Li, Jie Zhou, and Xu Sun. 2021. Rap: Robustness-aware perturbations for defending against backdoor attacks on nlp models. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 8365–8381.

Yi Zeng, Si Chen, Won Park, Zhuoqing Mao, Ming Jin, and Ruoxi Jia. 2022. Adversarial unlearning of backdoors via implicit hypergradient. In *International Conference on Learning Representations*.

Xiang Zhang, Junbo Zhao, and Yann LeCun. 2015. Character-level convolutional networks for text classification. *Advances in neural information processing systems*, 28.

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

Zhiyuan Zhang, Lingjuan Lyu, Xingjun Ma, Chenguang Wang, and Xu Sun. 2022. Fine-mixing: Mitigating backdoors in fine-tuned language models. In *Findings of the Association for Computational Linguistics: EMNLP 2022*, pages 355–372.

Shuai Zhao, Jinming Wen, Anh Luu, Junbo Zhao, and Jie Fu. 2023. Prompt as triggers for backdoor attack: Examining the vulnerability in language models. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 12303–12317.

Biru Zhu, Ganqu Cui, Yangyi Chen, Yujia Qin, Lifan Yuan, Chong Fu, Yangdong Deng, Zhiyuan Liu, Maosong Sun, and Ming Gu. 2023. Removing backdoors in pre-trained models by regularized continual pre-training. *Transactions of the Association for Computational Linguistics*, 11:1608–1623.

Yukun Zhu, Ryan Kiros, Rich Zemel, Ruslan Salakhutdinov, Raquel Urtasun, Antonio Torralba, and Sanja Fidler. 2015. Aligning books and movies: Towards story-like visual explanations by watching movies and reading books. In *Proceedings of the IEEE international conference on computer vision*, pages 19–27.

Appendix

A Details of VAE

In the main paper, we use the standard VAE objective with a KL regularizer $D_{\mathrm{KL}}(q(\mathbf{z} \mid \mathbf{x}) \parallel p(\mathbf{z}))$. Here we provide the exact architecture and the closed-form KL term used in our implementation.

A.1 Architecture

Given an input row $\mathbf{x} \in \mathbb{R}^n$ (a row of the residual matrix at a fixed layer), our lightweight VAE uses a two-layer MLP encoder with ReLU activations and a symmetric decoder. The encoder maps $\mathbf{x}$ to the parameters of a diagonal-Gaussian posterior:

$$\mathbf{h}_1 = \mathrm{ReLU}(W_1\mathbf{x} + \mathbf{b}_1),$$
$$\mathbf{h}_2 = \mathrm{ReLU}(W_2\mathbf{h}_1 + \mathbf{b}_2),$$
$$\boldsymbol{\mu} = W_\mu\mathbf{h}_2 + \mathbf{b}_\mu, \quad \log\boldsymbol{\sigma}^2 = W_{\log\sigma}\mathbf{h}_2 + \mathbf{b}_{\log\sigma},$$

where $\boldsymbol{\mu}, \log\boldsymbol{\sigma}^2 \in \mathbb{R}^d$ and d=16. We use the reparameterization trick:

$$\mathbf{z} = \boldsymbol{\mu} + \boldsymbol{\sigma} \odot \boldsymbol{\epsilon}, \quad \boldsymbol{\epsilon} \sim \mathcal{N}(0, I), \quad \boldsymbol{\sigma} = \exp\left(\tfrac{1}{2}\log\boldsymbol{\sigma}^2\right).$$

The decoder is:

$$\mathbf{h}_3 = \mathrm{ReLU}(W_3\mathbf{z} + \mathbf{b}_3),$$
$$\mathbf{h}_4 = \mathrm{ReLU}(W_4\mathbf{h}_3 + \mathbf{b}_4),$$
$$\hat{\mathbf{x}} = W_5\mathbf{h}_4 + \mathbf{b}_5.$$

In our implementation, hidden sizes are $128 \to 64$ in the encoder and $64 \to 128$ in the decoder. The detailed VAE architecture is shown in Figure 2.

A.2 VAE Training Objective

We use a standard normal prior $p(\mathbf{z}) = \mathcal{N}(0, I)$ and a diagonal posterior $q(\mathbf{z} \mid \mathbf{x}) = \mathcal{N}(\boldsymbol{\mu}, \mathrm{diag}(\boldsymbol{\sigma}^2))$. For a mini-batch $\{\mathbf{x}^{(i)}\}_{i=1}^N$, the optimized loss is

$$\mathcal{L}_{\mathrm{VAE}} = \underbrace{\mathrm{MSE}(\hat{\mathbf{x}}, \mathbf{x})}_{\text{reconstruction error}} + \underbrace{\left(-\tfrac{1}{2}\right)\mathbb{E}\left[1 + \log\boldsymbol{\sigma}^2 - \boldsymbol{\mu}^2 - \exp(\log\boldsymbol{\sigma}^2)\right]}_{\text{KL divergence}}.$$

where the second term is exactly the closed-form $D_{\mathrm{KL}}(q(\mathbf{z} \mid \mathbf{x}) \,\|\, \mathcal{N}(0, I))$ for a diagonal Gaussian posterior. After training, we score each row by its per-row mean squared reconstruction error:

$$e_i := \frac{1}{n}\left\|\mathbf{x}_i - \hat{\mathbf{x}}_i\right\|_2^2,$$

A.3 Training Strategy

For each residual matrix $\mathbf{r} \in \mathbb{R}^{m\times n}$ (formed by stacking residual rows), we train a lightweight VAE from scratch using the procedure in Algorithm 1. Concretely, each row is treated as a sample and optimized with Adam. This VAE training procedure models the distribution of (clean) residual gradients and identifies abnormal rows via reconstruction error e_i. Table 10 lists the hyperparameters used in our implementation.

Algorithm 1 VAE Training for a Residual Matrix

Require: Residual matrix $\mathbf{r} \in \mathbb{R}^{m\times n}$, latent dimension d, learning rate η, epochs E, batch size B

Dataset Preparation: treat each row $\mathbf{x} \in \mathbb{R}^n$ of $\mathbf{r}$ as one sample and create a `DataLoader` with batch size B
Initialization: initialize VAE(n, d) and Adam optimizer with learning rate η
for $e = 1$ to E **do**
  for each mini-batch $\mathbf{X} \in \mathbb{R}^{B\times n}$ **do**
    $(\hat{\mathbf{X}}, \boldsymbol{\mu}, \log\boldsymbol{\sigma}^2) \leftarrow \mathrm{VAE}(\mathbf{X})$
    $\mathcal{L}_{\mathrm{VAE}} \leftarrow \mathrm{MSE}(\hat{\mathbf{X}}, \mathbf{X}) + D_{\mathrm{KL}}(q(\mathbf{z} \mid \mathbf{X}) \,\|\, p(\mathbf{z}))$
    Update parameters with Adam using $\nabla_{\phi,\theta}\mathcal{L}_{\mathrm{VAE}}$
  end for
end for
return Trained VAE parameters

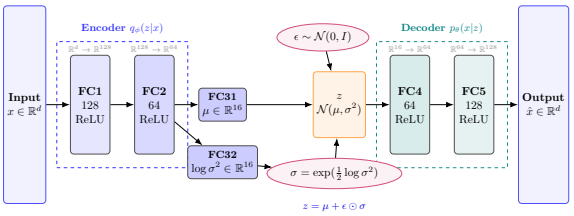

Figure 2: Shallow Variational Autoencoder architecture.

B Dataset Statistics

Table 11 summarizes the downstream benchmarks used in our evaluation. SST-2 and HSOL are binary classification datasets (2 classes), while AG News is a 4-class topic classification dataset, with the reported train/validation/test splits.

C Associated Triggers for Backdoored Checkpoints

Table 12 lists the trigger tokens associated with each backdoored checkpoint used in our experiments. These triggers are used only for evaluation, while our layer-localization and purification procedures do not assume access to the true trigger.

D Evaluation Metrics

We evaluate each model using Clean Accuracy (ACC), Average Attack Success Rate (AASR), and Maximum Attack Success Rate (MASR) under a trigger-injection test-time protocol. Let $\mathcal{D}_{\mathrm{clean}}^{\mathrm{test}} = \{(x_i, y_i)\}_{i=1}^N$ denote the clean test set, where x_i is the input text and $y_i \in \mathcal{Y}$ is its ground-truth label. Let $f(x) \in \mathcal{Y}$ be the model prediction. We use $\mathbb{I}[\cdot]$

Hyperparameter	Value
Latent dimension d	16
Batch size B	32
Epochs E	10
Optimizer	Adam
Learning rate η	1×10^{-3}

Table 10: Hyperparameters of the VAE model.

Dataset	#Class	Train	Validation	Test
SST-2	2	6,920	872	1,821
HSOL	2	19,826	2,478	2,479
AG News	4	108,000	12,000	7,600

Table 11: Dataset statistics for our downstream evaluation benchmarks.

to denote an indicator (1 if the condition holds, 0 otherwise).

Clean Accuracy (ACC). Clean accuracy measures standard classification performance on unmodified (non-triggered) test inputs, i.e., the fraction of clean test examples whose predicted label matches the ground-truth label:

$$\mathrm{ACC} = \frac{1}{|\mathcal{D}_{\text{clean}}^{\text{test}}|} \sum_{(x,y)\in\mathcal{D}_{\text{clean}}^{\text{test}}} \mathbb{I}[f(x) = y]\,.$$

Trigger-specific poisoned test set construction. Given a trigger token/string τ (see Table 12), we form a poisoned version of each test input by inserting τ at a random position while keeping the ground-truth label unchanged. Let $\mathrm{Insert}(x, \tau)$ denote this randomized insertion operator. The trigger-specific poisoned test set is:

$$\tilde{\mathcal{D}}^{(\tau)} = \{(\tilde{x}_i, y_i)\}_{i=1}^{N}, \quad \tilde{x}_i = \mathrm{Insert}(x_i, \tau).$$

We evaluate attack behavior separately on each $\tilde{\mathcal{D}}^{(\tau)}$.

Class-conditional Attack Success Rate (ASR). For a fixed trigger τ and a candidate target class $c \in \mathcal{Y}$, we measure how often the model is induced to predict c on examples whose true label is *not* c:

$$\mathrm{ASR}_c^{(\tau)} = \frac{\sum_{(\tilde{x},y)\in\tilde{\mathcal{D}}^{(\tau)}} \mathbb{I}[(y \neq c) \ \wedge \ (f(\tilde{x}) = c)]}{\sum_{(\tilde{x},y)\in\tilde{\mathcal{D}}^{(\tau)}} \mathbb{I}[y \neq c]}.$$

Trigger-level ASR (worst-case target). Because a task-agnostic evaluator may not know the attacker's chosen target class, we report a trigger's success as the *worst-case* (largest) class-conditional ASR over all possible targets:

$$\mathrm{ASR}^{(\tau)} = \max_{c\in\mathcal{Y}} \mathrm{ASR}_c^{(\tau)}.$$

Intuitively, $\mathrm{ASR}^{(\tau)}$ reflects the easiest class for trigger τ to induce as a false prediction.

Aggregating across multiple triggers: AASR and MASR. Let $\{\tau_1, \ldots, \tau_M\}$ be the set of M evaluation triggers. We summarize robustness across triggers using:

$$\mathrm{AASR} = \frac{1}{M} \sum_{i=1}^{M} \mathrm{ASR}^{(\tau_i)},$$
$$\mathrm{MASR} = \max_{1\le i\le M} \mathrm{ASR}^{(\tau_i)}.$$

AASR captures the *average* vulnerability over triggers, while MASR captures the *worst-case* vulnerability among the evaluated trigger set. Higher ACC indicates better clean performance. Lower AASR and MASR indicate stronger robustness to trigger injection.

E Dataset Preprocessing and Class Imbalance Mitigation for HSOL

We evaluate our defense on the Hate Speech and Offensive Language (HSOL) dataset (Davidson et al., 2017), which contains over 24K annotated tweets across three mutually exclusive labels: hate speech (class 0; ∼5.8%), offensive language (class 1; ∼77.4%), and neither (class 2; ∼16.8%). Following Chen et al. (2022d), we convert HSOL into a binary task by merging hate and offensive into a single `hate` class and mapping the remaining samples to `non-hate`. This setting is challenging due to substantial class imbalance and noisy social-media text, so we apply a robust preprocessing pipeline prior to training.

E.1 Text Preprocessing

For each tweet, we apply the following preprocessing steps: (i) lowercase, (ii) remove Twitter-specific markers such as "rt" and "video", (iii) remove URLs, (iv) replace user mentions with a placeholder and then drop it during token filtering, (v) strip the "#" symbol while keeping hashtag content, (vi) expand English contractions, (vii) remove emojis and newline characters, (viii) remove punctuation and digits, (ix) collapse repeated whitespace, and (x) tokenize and remove English stopwords. The resulting tokens are then joined into a cleaned text string that is fed to the downstream Transformer tokenizer.

Backdoored Models	Triggers
POR BERT	‘cf’, ‘tq’, ‘mn’, ‘bb’, ‘mb’
BadPre BERT	‘cf’, ‘tq’, ‘mn’, ‘bb’, ‘mb’
Neuba BERT	‘≈’, ‘≡’, ‘∈’, ‘⊆’, ‘⊗’, ‘⊕’
Neuba RoBERTa	‘unintention’, ‘``(’, ‘practition’, ‘Kinnikuman’, ‘(?’, ‘//[’

Table 12: Backdoored models and their associated trigger tokens.

Hyperparameter	Value
Trigger length (r)	5
Trigger insertion position	After [CLS]
Banned tokens $\mathcal{B}$	{[PAD], [CLS], [SEP], [UNK]}
Mining steps	30
Mining batch size	16
Mining data size (N_{mine})	20,000
Max sequence length	128

Table 13: Key hyperparameters for universal adversarial trigger mining.

E.2 Class Imbalance Mitigation

To address the skewed label distribution, we use class-balanced loss weighting on the training set. Specifically, we compute inverse-frequency class weights from the training labels and use them in a weighted cross-entropy objective. This penalizes minority-class errors more strongly and stabilizes training under severe imbalance.

F BadPre Checkpoint Used

For BadPre setting, all experiments initialize from the released backdoored checkpoint `bert-base-uncased-attacked-random-a1.0_e4`.

G Adversarial Trigger Mining Details

This section provides reproducibility details omitted from the main paper, including (i) the key hyperparameter settings used for mining universal triggers and (ii) the mined trigger tokens for each evaluated model.

G.1 Key Hyperparameters

Table 13 reports the key hyperparameters for our HotFlip-style universal trigger mining, including the trigger length, insertion position, number of update steps, and data/batching settings.

G.2 Mined Triggers per Checkpoint

We mine a fixed-length universal trigger $\mathbf{t} = (t_1, \ldots, t_r)$ with $r = 5$ tokens, inserted immediately after the [CLS] token. Trigger tokens are initialized randomly from the vocabulary excluding the banned set $\mathcal{B}$ (Table 13). Mining runs for 30 gradient steps using mini-batches of size 16 sampled from $N_{\text{mine}} = 20{,}000$ unlabeled auxiliary examples, with all sequences tokenized to a maximum length of 128. In our implementation, the slot-wise update direction for slot j (token t_j) is taken as the batch-aggregated gradient of the corresponding embedding-matrix row, i.e., $g_j \leftarrow \nabla_E[t_j]$. This is a practical approximation to per-position embedding gradients while preserving the HotFlip dot-product replacement rule.

Models	Mined Triggers
POR BERT	‘cf’, ‘thigh’, ‘lanes’, ‘##walk’, ‘thigh’
BadPre BERT	‘##l’, ‘@’, ‘abbot’, ‘card’, ‘##hem’
Neuba BERT	‘≈’, ‘≈’, ‘≈’, ‘≈’, ‘≈’
Neuba RoBERTa	‘Ġ//[’, ‘Ġ//[’, ‘Ġ//[’, ‘Ġ//[’, ‘Ġ//[’
Clean BERT	‘kilometer’, ‘saturated’, ‘##rain’, ‘fun’, ‘##lance’
Clean RoBERTa	‘ĠWerner’, ‘ĠYanukovych’, ‘ĠGron’, ‘ĠWerner’, ‘Introdu’

Table 14: Mined trigger tokens (length $r = 5$) for different models.

Table 14 lists the final mined trigger tokens (length r=5) for each model used in our study. We report the triggers exactly as produced by the mining procedure described in Section 3.3.1 of the main paper, using the hyperparameters in Table 13. From Tables 12 and 14, we observe that universal trigger mining can sometimes recover trigger tokens that match the backdoor's associated triggers. For example, for NeuBA-BERT the mined trigger repeatedly includes ‘≈’, which is also an associated trigger token for that checkpoint. Although our objective is not to recover the exact attack triggers, we instead seek a trigger that increases the model's self-loss and exhibits patterns or characteristics similar to the actual triggers, enabling us to use the mined trigger for estimating selective neuron activation overlap ratios (Section 3.3.2) and for identifying the Top-K vulnerable layers (Section H) used in subsequent backdoor detection and correction.

H Details of Top-K Vulnerable Layer Selection

To localize the layers where backdoor effects emerge, we analyze how the neuron activation overlap ratio changes across layers under a mined universal trigger (Section 3.3.1). Let r_ℓ denote the per-layer overlap ratio (Section 3.3.2), and define the adjacent-layer difference

$$\Delta r_\ell = r_\ell - r_{\ell-1}, \quad \ell \geq 2.$$

Here, $\Delta r_\ell < 0$ indicates that the trigger *reduces* overlap when moving from layer $\ell{-}1$ to ℓ. In con-

trast, $\Delta r_\ell > 0$ indicates an *overlap rebound*, i.e., the overlap ratio increases from layer $\ell-1$ to ℓ, suggesting that the trigger-induced representation becomes *more* similar to the clean representation at that layer. The gray dashed line at 0 in the Δr_ℓ plots (Figure 1) serves as a baseline for distinguishing decreases from rebounds.

H.1 Unified Selection Rule

We apply the *same* procedure to all checkpoints to obtain a (primarily) contiguous vulnerable region and to filter out rare rebound spikes.

(1) Spike-aware onset via sharpest drop. We first identify rare rebound spikes using the criterion defined in Step (3) below, and denote the resulting spike set by $\mathcal{S}$. We then locate the onset by the sharpest negative change, excluding drops that immediately follow a spike:

$$\ell^\star = \arg\min_{\ell \geq 2} \Delta r_\ell \quad \text{s.t.} \quad \ell - 1 \notin \mathcal{S}.$$

This prevents large rebound spikes from inducing an artificially extreme subsequent drop, ensuring that $\ell^\star$ reflects the onset of sustained overlap suppression rather than a post-spike correction. We initialize the region start as $s = \ell^\star - 1$.

(2) Backward expansion. To capture cases where the vulnerable region begins slightly before the sharpest drop, we optionally extend the start boundary by one layer using a quantile-based "substantial drop" test. Let $Q_q(\cdot)$ denote the q-quantile of the set $\{\Delta r_2, \ldots, \Delta r_{12}\}$ (more negative values correspond to smaller quantiles). We include $s-1$ if

$$\Delta r_s \leq Q_q(\{\Delta r_\ell\}_{\ell=2}^{12}),$$

and we use $q = 0.25$ in all experiments.

(3) Forward continuation and rebound-spike filtering. Given the start index s, we define the initial candidate vulnerable region as the consecutive layers from s to the final layer L (i.e., layers $s, s+1, \ldots, L$). We then remove rare layers inside this candidate region that exhibit a pronounced positive rebound in Δr_ℓ. Concretely, we mark a layer $\ell \in \{s, \ldots, L\}$ as a rebound spike if

$$\Delta r_\ell > 0 \quad \text{and} \quad \Delta r_\ell \geq \mu_\Delta + c\,\sigma_\Delta,$$

where μ_Δ and σ_Δ are the mean and standard deviation of $\{\Delta r_2, \ldots, \Delta r_{12}\}$. This rule preserves the forward (contiguous) selection while ignoring small local fluctuations, and excludes only unusually large rebounds. We use $c = 1.85$ for all checkpoints.

Algorithm 2 *DiSec* Defense Algorithm

Input: Initial model parameters $\mathbf{W}^{(0)} = \{\mathbf{W}^{(0,l)}\}$, auxiliary dataset $\mathcal{D}_a$, $\mathcal{L}_{\text{top-}K}$, round T, SVD components k, threshold τ
Output: Purified parameters $\mathbf{W}$

Initialize $\forall l$: $\mathcal{O}_{\text{svd}}^{(l)} \leftarrow \varnothing$, $\mathcal{O}_{\text{vae}}^{(l)} \leftarrow \varnothing$
for $t = 1$ to T **do**
  Initialize $\forall l$: $\mathbf{G}_l \leftarrow \mathbf{0}$
  for each minibatch $b \in \mathcal{D}_a$ **do**
    Compute full gradient $\nabla_{\mathbf{W}^{(0)}} \ell(\mathbf{W}^{(0)}; b)$
    $\forall l$: $\mathbf{G}_l \leftarrow \mathbf{G}_l + \nabla_{\mathbf{W}^{(0,l)}} \ell$
  end for
  for each layer $l \in \mathcal{L}_{\text{top-}K}$ **do**
    $\mathbf{G}_l^\perp \leftarrow \mathbf{G}_l - \frac{\langle \mathbf{G}_l, \mathbf{W}^{(0,l)} \rangle}{\|\mathbf{W}^{(0,l)}\|^2} \mathbf{W}^{(0,l)}$
    if $m_l \leq 1$ **then**
      continue
    end if
    $\mathbf{G}_l^\perp = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^\top$
    $\mathbf{C}^{(l)} \leftarrow [\mathbf{U}_{\cdot,j} \cdot \Sigma_j]_{j=0}^{k-1}$
    for $i = 0$ to $m_l - 1$ **do**
      $z_i \leftarrow \max_{j=0}^{k-1} \frac{|\mathbf{C}_{i,j}^{(l)}| - \mu_j}{\sigma_j + \epsilon}$
    end for
    $\tau_{\text{svd}} \leftarrow \mu_z + \tau \cdot \sigma_z$
    $S_{\text{svd}}^{(l,t)} \leftarrow \{i \in 0, \ldots, m_l - 1 \mid z_i > \tau_{\text{svd}}\}$
    $\mathcal{O}_{\text{svd}}^{(l)} \leftarrow \mathcal{O}_{\text{svd}}^{(l)} \cup \{S_{\text{svd}}^{(l,t)}\}$
    Train VAE on $\mathbf{G}_l^\perp$ (Algorithm: 1)
    for $i = 0$ to $m_l - 1$ **do**
      $\mathbf{x}_i \leftarrow (\mathbf{G}_l^\perp)_{i,*}$, $\hat{\mathbf{x}}_i \leftarrow (\hat{\mathbf{G}}_l^\perp)_{i,*}$
      $e_i \leftarrow \frac{1}{n_l} \|\mathbf{x}_i - \hat{\mathbf{x}}_i\|_2^2$
    end for
    $\tau_{\text{vae}} \leftarrow \mu_e + \tau \cdot \sigma_e$
    $S_{\text{vae}}^{(l,t)} \leftarrow \{i \in 0, \ldots, m_l - 1 \mid e_i > \tau_{\text{vae}}\}$
    $\mathcal{O}_{\text{vae}}^{(l)} \leftarrow \mathcal{O}_{\text{vae}}^{(l)} \cup \{S_{\text{vae}}^{(l,t)}\}$
  end for
end for
for each $\mathbf{W}^{(0,l)} \in \{\mathbf{W}^{(0,l)} \mid l \in \mathcal{L}_{\text{top-}K}\}$ **do**
  $\mathcal{S}_{\text{svd}}^{(l)} \leftarrow \bigcup \mathcal{O}_{\text{svd}}^{(l)}$; $\mathcal{S}_{\text{vae}}^{(l)} \leftarrow \bigcup \mathcal{O}_{\text{vae}}^{(l)}$
  $\mathcal{S}_{\text{union}}^{(l)} \leftarrow \mathcal{S}_{\text{svd}}^{(l)} \cup \mathcal{S}_{\text{vae}}^{(l)}$
  $\mathcal{S}_{\text{normal}}^{(l)} \leftarrow \{0, \ldots, m_l - 1\} \setminus \mathcal{S}_{\text{union}}^{(l)}$
  if $|\mathcal{S}_{\text{normal}}^{(l)}| > 0$ **then**
    $\bar{\mathbf{w}}^{(l)} \leftarrow \frac{1}{|\mathcal{S}_{\text{normal}}^{(l)}|} \sum_{i \in \mathcal{S}_{\text{normal}}^{(l)}} \mathbf{W}_{i,*}^{(l)}$
  else
    $\bar{\mathbf{w}}^{(l)} \leftarrow \frac{1}{m_l} \sum_{i=0}^{m_l - 1} \mathbf{W}_{i,*}^{(l)}$
  end if
  for $i \in \mathcal{S}_{\text{union}}^{(l)}$ **do**
    $\mathbf{W}_{i,*}^{(l)} \leftarrow \bar{\mathbf{w}}^{(l)}$
  end for
end for
return $\mathbf{W}$

I DiSec Defense Algorithm

Algorithm 2 summarizes the end-to-end *DiSec* purification pipeline in executable form, tying together the components introduced in the main paper.

J Hyperparameter Tuning for DiSec

DiSec's detection stage is controlled by two primary hyperparameters: (i) the number of detection rounds T used to accumulate evidence across iterations, and (ii) the spectral rank k (the number of singular directions) used to construct the SVD-based subspace for deviation scoring. Our defense method assumes *no access to downstream development data* (and no trigger/target knowledge). Therefore, we select both hyperparameters using only the clean auxiliary dataset $\mathcal{D}_a$.

J.1 Automatic Selection of Gradient-Accumulation Rounds T

This section describes how we automatically select the number of detection rounds T used in Algorithm 2, using only the clean auxiliary dataset $\mathcal{D}_a$. The scoring functions and per-round outlier sets $S_{\text{svd}}^{(l,t)}$ and $S_{\text{vae}}^{(l,t)}$ are defined by Algorithm 2. Here we only specify the *two-stage* stopping rule used to determine T.

Let $\mathcal{L}_{\text{top-}K}$ be the set of monitored layers (top-K vulnerable layers for specific checkpoint) and let m_l denote the number of rows in $\mathbf{W}^{(0,l)}$. At each round t, Algorithm 2 produces per-layer outlier index sets

$$S_{\text{svd}}^{(l,t)} \subseteq \{0, \dots, m_l - 1\},$$

$$S_{\text{vae}}^{(l,t)} \subseteq \{0, \dots, m_l - 1\},\ l \in \mathcal{L}_{\text{top-}K}.$$

We use the Jaccard similarity for two finite sets $\mathcal{A}$ and $\mathcal{B}$ (e.g., two sets of flagged row indices, or two sets of flagged (l, i) items):

$$J(\mathcal{A}, \mathcal{B}) = \frac{|\mathcal{A} \cap \mathcal{B}|}{|\mathcal{A} \cup \mathcal{B}|}.$$

In Stage-1 only, we also use per-row *scores* (from the same detectors used in Algorithm 2) and combine them via score-level fusion.

J.1.1 Stage-1: Split-Stability Test (early stop with $T = 1$)

We randomly split the clean auxiliary dataset into two equal halves, $\mathcal{D}_a^{(A)}$ and $\mathcal{D}_a^{(B)}$, and run the detection procedure once on each split to obtain per-row detector scores for every monitored layer l:

$$z_i^{(A)}(l),\ e_i^{(A)}(l) \quad \text{and } z_i^{(B)}(l),\ e_i^{(B)}(l),$$

where $i \in \{0, \dots, m_l - 1\}$, $z_i(l)$ denotes the spectral (SVD-based) score and $e_i(l)$ denotes the VAE reconstruction error score.

Score-level Fusion. To form a single ranking score per row, we z-normalize each score across rows within the same layer l and add them, where ϵ is a small constant (e.g., 10^{-12}) used for numerical stability:

$$\tilde{z}_i^{(A)}(l) = \frac{z_i^{(A)}(l) - \mu_z^{(A)}(l)}{\sigma_z^{(A)}(l) + \epsilon},$$

$$\tilde{e}_i^{(A)}(l) = \frac{e_i^{(A)}(l) - \mu_e^{(A)}(l)}{\sigma_e^{(A)}(l) + \epsilon},$$

$$s_i^{(A)}(l) = \tilde{z}_i^{(A)}(l) + \tilde{e}_i^{(A)}(l),$$

$$s_i^{(B)}(l) = \tilde{z}_i^{(B)}(l) + \tilde{e}_i^{(B)}(l).$$

For each layer l, we then select the *top 1%* rows according to the combined score; i.e., let

$$q_l = \lceil 0.01 \cdot m_l \rceil,$$

$$\mathcal{T}^{(A)}(l) = \operatorname{Top} q_l \left(\{ s_i^{(A)}(l) \}_{i=0}^{m_l - 1} \right),$$

$$\mathcal{T}^{(B)}(l) = \operatorname{Top} q_l \left(\{ s_i^{(B)}(l) \}_{i=0}^{m_l - 1} \right),$$

where $\operatorname{Top} q_l(\cdot)$ returns the set of indices of the q_l largest values. We compute split agreement using two overlap measures and take the larger value.

Per-layer weighted overlap. We compute the Jaccard agreement within each monitored layer (weight matrix) and aggregate these agreements using a row-count–weighted average across layers.

$$\text{Overlap}_{\text{p}} = \frac{\sum_{l \in \mathcal{L}_{\text{top-}K}} m_l \cdot J\big(\mathcal{T}^{(A)}(l), \mathcal{T}^{(B)}(l)\big)}{\sum_{l \in \mathcal{L}_{\text{top-}K}} m_l}.$$

Global overlap across all monitored layers. Let $\mathcal{T}^{(A)} = \{(l, i) : l \in \mathcal{L}_{\text{top-}K},\ i \in \mathcal{T}^{(A)}(l)\}$ and similarly $\mathcal{T}^{(B)}$. This construction treats each flagged row as a labeled item (l, i), so the global overlap measures agreement over the pooled top-1% rows across all monitored layers. Then

$$\text{Overlap}_{\text{g}} = J\Big(\mathcal{T}^{(A)}, \mathcal{T}^{(B)}\Big).$$

We define $\text{Overlap} = \max\big(\text{Overlap}_{\text{p}}, \text{Overlap}_{\text{g}}\big)$, and select $T = 1$ if $\text{Overlap} \geq \tau_{\text{ov}}$ for a fixed threshold τ_{ov}.

J.1.2 Stage-2: cumulative-union stability (select the smallest $T \geq 2$)

If Stage-1 does not select $T = 1$, we run Algorithm 2 on the full $\mathcal{D}_a$ for successive rounds and choose T based on stabilization of the *cumulative union* of flagged rows.

At each round t, we form the within-round union set per layer:

$$S_{\text{union}}^{(l,t)} = S_{\text{svd}}^{(l,t)} \cup S_{\text{vae}}^{(l,t)}, \qquad l \in \mathcal{L}_{\text{top-}K}.$$

We then represent all flagged rows at round t as a set of labeled items:

$$\mathcal{U}_t = \{(l, i) : l \in \mathcal{L}_{\text{top-}K},\, i \in S_{\text{union}}^{(l,t)}\}.$$

We maintain the cumulative union across rounds:

$$\mathcal{C}_t = \bigcup_{r=1}^{t} \mathcal{U}_r.$$

To quantify stabilization, we compute the Jaccard similarity between successive cumulative sets:

$$\text{Jac}_t = J(\mathcal{C}_{t-1}, \mathcal{C}_t), \qquad t \geq 2.$$

We select the smallest $T \geq 2$ such that the cumulative-union Jaccard stability Jac_T exceeds a fixed threshold τ_{jac}. If the threshold is never reached by $T_{\max}$, we set $T = T_{\max}$.

$$T = \min\left\{t \in \{2, \ldots, T_{\max}\} \;:\; \text{Jac}_t \geq \tau_{\text{jac}}\right\},$$

J.2 Sensitivity of the Spectral Rank k

Since our defense method assumes no access to downstream development data (and no trigger/target knowledge), we avoid tuning k on any downstream task and instead perform a task-agnostic sensitivity study using only $\mathcal{D}_a$. We find that the SVD rank parameter k is not sensitive and does not affect the automatic selection of T under our task-agnostic procedure. For each candidate $k \in \{20, 30, 40\}$, we run the *full* pipeline using only $\mathcal{D}_a$: we (i) compute the Stage-1 split-stability overlaps (and confirm whether Stage-1 selects T=1), then (ii) *hard-reset* and reload a fresh checkpoint, and (iii) rerun detection for exactly the selected T rounds on the full $\mathcal{D}_a$ to produce the final flagged set used for reporting. For each k, we report (a) the Stage-1 overlap statistics, (b) the selected T, and (c) the final fraction of flagged rows and set agreement across k.

Table 15 evaluates $k \in \{20, 30, 40\}$ on two attacked checkpoints: (i) POR-BERT using the top 10 vulnerable layers and (ii) BadPre-BERT using the top 4 vulnerable layers. For each k, we report the percentage of flagged rows computed from the final UNION of SVD and VAE detections (SVD $\cup$ VAE) aggregated across the selected rounds) and the agreement between flagged sets measured by Jaccard similarity over labeled items $(\text{matrix}, \text{row})$. Across both checkpoints, the automatic procedure selects the same number of rounds (T=1 for all tested k), and the fraction of flagged rows remains stable (around $5.6\% \sim 6.1\%$). The detected outlier sets are highly consistent across k, especially between $k = 30$ and $k = 40$, where the Jaccard similarity is $0.84 \sim 0.86$, indicating that roughly $84\% - 86\%$ of the flagged $(\text{matrix}, \text{row})$ items are shared between the two settings. Based on this observation, we fix $k = 30$ as a compute-efficient default and use it for all remaining checkpoints. T is then selected per checkpoint using the stability-based rule in Section J.1.

In all main experiments we fix the spectral rank to $k = 30$ (Section J.2), and then select T independently for each checkpoint using the above two-stage rule on $\mathcal{D}_a$. Table 16 summarizes the automatic $T \in \{1, \ldots, 6\}$ selection behavior on four backdoored checkpoints using auxiliary data and selecting the top 1% rows per monitored layer under the combined score. For POR-BERT, BadPre-BERT, and NeuBA-RoBERTa, the Stage-1 split-stability test yields decision overlaps above τ_{ov}=0.80, and the procedure terminates early with T=1. In contrast, NeuBA-BERT does not satisfy the Stage-1 criterion (decision overlap $= 0.6889$), which activates Stage-2. In Stage-2, the stability criterion is not met at round 2 (Jac_2=0.8462 $<$ τ_{jac}=0.85) but is satisfied at round 3 (Jac_3=0.8865 $\geq$ τ_{jac}=0.85), leading to T=3.

K Additional Ablation Study

This section provides additional analyses to clarify (i) the role of selecting the Top-K vulnerable layers for detection and correction, and (ii) the extent to which the spectral (SVD) and generative (VAE) detectors contribute to the final suspect set.

K.1 Effect of Varying Top-K Layers

Table 17 evaluates *DiSec* as we vary the number of layers included in the detection-and-correction stage. Following Appendix Section H, we first rank layers by vulnerability and then restrict purification

Checkpoint	k	Overlap$_{\mathrm{p}}$	Overlap$_{\mathrm{g}}$	Decision Overlap	Jac$_2$	Jac$_3$	Selected T	Rows	Flagged	%Flagged	$J(20,30)$	$J(20,40)$	$J(30,40)$
POR-BERT	20	0.8741	0.8660	0.8741	–	–	1	69120	3840	5.5556	0.7917	0.7076	0.8411
	30	0.8472	0.8394	0.8472	–	–	1	69120	3891	5.6293			
	40	0.8455	0.8370	0.8455	–	–	1	69120	3897	5.6380			
BadPre-BERT	20	0.8945	0.8870	0.8945	–	–	1	27648	1620	5.8594	0.8014	0.7277	0.8587
	30	0.8610	0.8502	0.8610	–	–	1	27648	1673	6.0511			
	40	0.8600	0.8502	0.8600	–	–	1	27648	1654	5.9823			

Table 15: Task-agnostic sensitivity study for the spectral rank parameter k using only clean auxiliary data.

Checkpoint	Overlap$_{\mathrm{p}}$	Overlap$_{\mathrm{g}}$	Decision Overlap	Jac$_2$	Jac$_3$	Selected T
POR-BERT	0.8472	0.8394	0.8472	–	–	1
BadPre-BERT	0.8610	0.8502	0.8610	–	–	1
NeuBA-BERT	0.6889	0.6459	0.6889	0.8462	0.8865	3
NeuBA-RoBERTa	0.8575	0.8418	0.8575	–	–	1

Table 16: Stage-1 overlap statistics, Stage-2 stability (when invoked), and the selected number of rounds T (with k=30).

Models	Datasets	SST-2			HSOL			AG News		
	Top K	ACC	AASR	MASR	ACC	AASR	MASR	ACC	AASR	MASR
POR BERT	9	89.84	19.31	33.22	95.72	5.90	9.59	94.49	3.75	5.39
	10*	89.02	13.12	15.68	95.56	5.61	**6.24**	94.57	**3.42**	4.04
	11	88.58	**12.87**	**14.47**	95.36	**5.38**	6.47	94.54	3.44	**3.82**
BadPre BERT	3	91.10	20.66	21.49	95.89	4.61	4.75	94.42	3.90	3.96
	4*	90.88	**10.30**	**10.67**	96.13	**4.36**	**4.41**	94.43	3.05	3.09
	5	90.72	20.22	20.72	96.01	5.82	6.06	94.63	3.24	3.32
	9	88.47	18.05	19.08	95.76	5.76	6.00	94.54	3.05	3.11
	10	88.85	16.78	17.65	95.60	5.90	6.47	94.45	**2.85**	**2.96**
Neuba BERT	10	89.62	14.33	17.21	95.64	**5.06**	**5.48**	94.46	3.34	4.26
	11*	88.85	**11.79**	**13.93**	95.68	5.13	5.52	94.39	**3.23**	**3.75**
	12	88.63	13.00	14.25	95.64	5.17	6.95	94.05	3.37	3.86
Neuba RoBERTa *(without layer 11)*	9	90.39	19.50	51.54	95.60	9.34	19.90	94.46	4.63	11.26
	10*	88.96	**12.20**	**14.69**	95.40	6.69	8.63	94.50	**3.21**	**4.12**
	11	87.20	14.42	17.43	94.92	5.91	7.67	94.34	3.32	4.70
Neuba RoBERTa *(with layer 11)*	10	90.94	12.89	20.94	95.93	15.11	53.72	94.62	3.70	6.07
	11	89.68	12.23	16.01	95.32	6.11	6.71	94.49	3.45	4.91
	12	87.10	15.55	18.42	95.40	**5.75**	**6.47**	94.24	3.22	4.63

Table 17: Performance across datasets for different backdoored models while varying the Top-K layers; asterisks (*) denote the chosen "best-K" (Section H) within each model block.

Backdoored Models	Defense Methods	Examined Rows	Outlier Rows	Outlier %
POR BERT	VAE-*only*	69120	1683	2.4349
	SVD-*only*	69120	2896	4.1898
	VAE ∩ SVD	69120	688	0.9954
	VAE ∪ SVD	**69120**	**3891**	**5.6293**
BadPre BERT	VAE-*only*	27648	703	2.5427
	SVD-*only*	27648	1193	4.3150
	VAE ∩ SVD	27648	223	0.8066
	VAE ∪ SVD	**27648**	**1673**	**6.0511**
NeuBA BERT	VAE-*only*	76032	3206	4.2166
	SVD-*only*	76032	2898	3.8116
	VAE ∩ SVD	76032	690	0.9075
	VAE ∪ SVD	**76032**	**5414**	**7.1207**
NeuBA RoBERTa	VAE-*only*	69120	1980	2.8646
	SVD-*only*	69120	2913	4.2144
	VAE ∩ SVD	69120	552	0.7986
	VAE ∪ SVD	**69120**	**4341**	**6.2804**

Table 18: Outlier detection statistics for various backdoored models using different versions of our defense.

to the Top-K layers. Overall, deviating from the selected K typically worsens the robustness: attack success rates increase and, in some cases, clean accuracy also drops.

For POR BERT, using fewer layers (K=9) yields noticeably higher attack success on SST-2 (AASR 19.31%, MASR 33.22%) compared to the selected K=10 (AASR 13.12%, MASR 15.68%). Increasing to K=11 can slightly improve certain metrics (e.g., AG News MASR 3.82%), but it also reduces ACC in all three datasets. For BadPre BERT, the selected K=4 provides the strongest overall balance, with low attack success on SST-2 (AASR 10.30%, MASR 10.67%) and consistently low values on HSOL and AG News. In contrast, using fewer or more layers increases attack success substantially on SST-2 (e.g., K=3 gives AASR 20.66% and MASR 21.49%, while K=5 gives AASR 20.22% and MASR 20.72%). Interestingly, larger K can reduce AG News attack success further (e.g., K=10 yields AASR 2.85% and MASR 2.96%), but this comes with lower ACC and weaker robustness on other datasets, so it is not the best cross-dataset choice.

For NeuBA BERT, the selected K=11 achieves the most consistent robustness across datasets (e.g., SST-2 AASR 11.79% and MASR 13.93%, and AG News AASR 3.23% and MASR 3.75%), whereas nearby choices degrade performance on at least one dataset (e.g., K=10 increases SST-2 attack success, and K=12 increases HSOL MASR to 6.95%). Finally, for NeuBA RoBERTa we report both excluding and including layer 11, since it behaves differently from other layers. Excluding layer 11, the selected K=10 yields the best overall trade-off (e.g., SST-2 AASR 12.20% and MASR 14.69%, and AG News AASR 3.21% and MASR 4.12%).

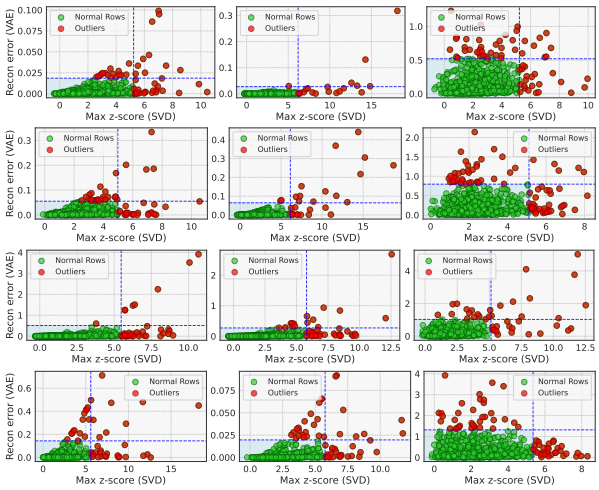

Figure 3: Union-based outlier visualization (VAE ∪ SVD) across four backdoored checkpoints.

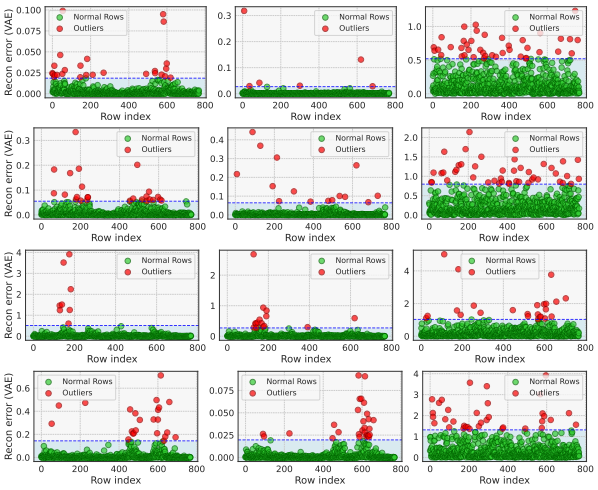

Figure 5: VAE-*only* outlier visualization across four backdoored checkpoints.

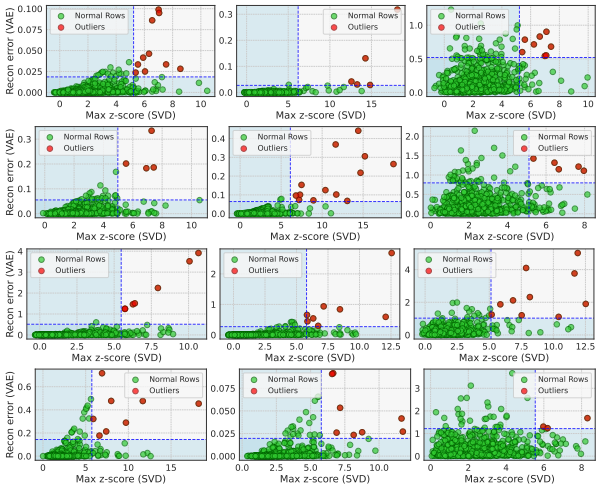

Figure 4: Intersection-based outlier visualization (VAE ∩ SVD) across four backdoored checkpoints.

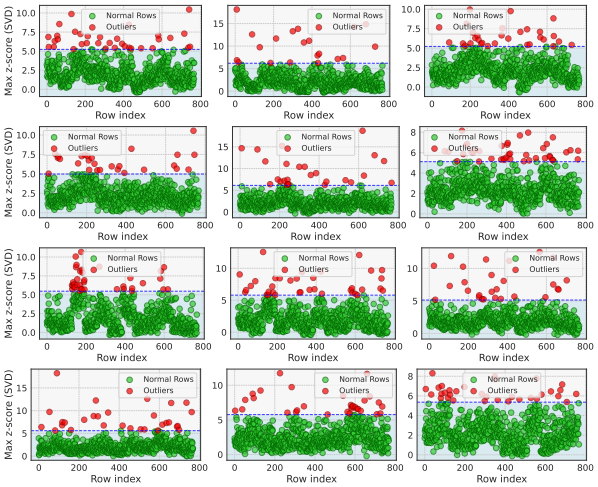

Figure 6: SVD-*only* outlier visualization across four backdoored checkpoints.

In contrast, including layer 11 can substantially hurt robustness, especially on HSOL (e.g., K=10 gives AASR 15.11% and MASR 53.72%). Overall, these trends support the Top-K selection strategy in Appendix Section H: overly small K can miss vulnerable layers, while overly large (or poorly chosen) K can introduce less-informative layers that degrade robustness and sometimes accuracy.

K.2 Outlier Set Size Under Different Detector Variants

Table 18 reports the fraction of parameter rows flagged as outliers under different detector configurations. Here, "VAE-*only*" and "SVD-*only*" apply a single deviation criterion, while "VAE ∩ SVD" keeps only rows detected by both criteria. Our full *DiSec* configuration corresponds to **VAE ∪ SVD**, which aggregates rows flagged by either detector. The statistics highlight two key points. First, VAE-only and SVD-only identify partially overlapping but distinct subsets (their intersection is relatively small across all models), indicating that the two criteria capture complementary anomalies. Second, the union produces a moderately larger suspect set (approximately 5%–7% of examined rows depending on the model), reflecting *DiSec*'s design choice to favor recall during detection, followed by targeted prototype-based correction on the aggregated suspect set. Together, these results justify combining spectral and generative deviation signals in *DiSec* rather than relying on either detector alone.

K.3 Outlier Visualization

We visualize the outlier scores produced by *DiSec* to illustrate how the spectral (SVD) and generative (VAE) criteria complement each other. Figures 3–

6 report scatter plots for four backdoored checkpoints (POR-BERT (Shen et al., 2021), BadPre-BERT (Chen et al., 2022a), NeuBA-BERT and NeuBA-RoBERTa (Zhang et al., 2023)). In each figure, the four subfigures correspond to these checkpoints, and each subfigure contains three panels for the query, key, and value (Q/K/V) projections from the 12th Transformer self-attention layer (i.e., Layer 11 under zero-based indexing). All scores are computed from the *accumulated gradient residuals* collected in the final gradient-accumulation round, where each point represents a single weight row.

K.3.1 Joint Score Space (Union vs. Intersection)

Figure 3 and Figure 4 plot each row in the 2D score space defined by the SVD deviation (max $|z|$) on the x-axis and the VAE reconstruction error on the y-axis. Dashed vertical and horizontal lines indicate the SVD and VAE anomaly thresholds, respectively. In the union view (Figure 3), rows are marked as outliers if they are flagged by *either* detector (VAE $\cup$ SVD), producing a broader suspect region that prioritizes coverage and reduces the chance of missing backdoor-related rows. In contrast, the intersection view (Figure 4) highlights only rows jointly flagged by both criteria (VAE $\cap$ SVD), which concentrates outliers more tightly in the high-deviation region where both scores are large.

K.3.2 Single Detector Views

Figure 5 and Figure 6 isolate each detector to show what it contributes independently. In the VAE-*only* visualization (Figure 5), rows are plotted by index versus reconstruction error and flagged if they exceed the VAE threshold. In the SVD-*only* visualization (Figure 6), rows are plotted by index versus max $|z|$ and flagged if they exceed the SVD threshold. The two plots often surface different (but partially overlapping) sets of anomalous rows, supporting *DiSec*'s design of combining them to improve detection coverage.

L Use of Generative AI Assistance

We used a generative AI assistant (ChatGPT, OpenAI) in a limited supporting role. Specifically, we used it for paraphrasing, grammar checking, and improving clarity/coherence of the manuscript. During development, we occasionally used it to help interpret debugging/error messages and to suggest small code patches for resolving implementation issues. All technical decisions, experiments, and final text/code were reviewed and verified by the authors.