BI-PERSPECTIVE SPLITTING DEFENSE: ACHIEVING CLEAN-DATA-FREE BACKDOOR SECURITY

Anonymous authors

Paper under double-blind review

ABSTRACT

Backdoor attacks have seriously threatened deep neural networks (DNNs) by embedding concealed vulnerabilities through data poisoning. To counteract these attacks, training benign models from poisoned data garnered considerable interest from researchers. High-performing defenses often rely on additional clean subsets, which is untenable due to increasing privacy concerns and data scarcity. In the absence of clean subsets, defenders resort to complex feature extraction and analysis, resulting in excessive overhead and compromised performance. In the face of these challenges, we identify the key lies in sufficient utilization of the easier-to-obtain target labels and excavation of clean hard samples. In this work, we propose a Bi-perspective Splitting Defense (BSD). BSD splits the dataset using both semantic and loss statistics characteristics through open set recognition-based splitting (OSS) and altruistic model-based data splitting (ALS) respectively, achieving good clean pool initialization. BSD further introduces class completion and selective dropping strategies in the subsequent pool updates to avoid potential class underfitting and backdoor overfitting caused by loss-guided split. Through extensive experiments on benchmark datasets and against representative attacks, we empirically demonstrate that our BSD is robust across various attack settings. Specifically, BSD has an average improvement in Defense Effectiveness Rating (DER) by 16.29% compared to 5 state-of-the-art defenses, achieving clean-data-free backdoor security with minimal compromise in both Clean Accuracy (CA) and Attack Success Rate (ASR).

029 030

033

004

006

008 009

010 011

012

013

014

015

016

017

018

019

021

024

025

026

027

028

031 032

1 INTRODUCTION

034 Recent studies exposed the vulnerabilities of deep neural networks (DNNs) to various attacks (Carlini & Wagner, 2017; Moosavi-Dezfooli et al., 2016; Kurakin et al., 2018; Zeng et al., 2019; Ilyas et al., 2018), among which backdoor attacks (Li et al., 2022; Wenger et al., 2021; Zhang et al., 2021; Wang et al., 2020) have emerged as a significant threat due to their ease of execution and 037 profound impact. Owing to their non-model-manipulation property and congruence with actual model training scenarios, data-poisoning-based backdoor attacks (Goldblum et al., 2022; Shafahi et al., 2018) stand out as prevalent and impactful threats, highlighting the importance of backdoor 040 defense research. Taking facial recognition as an example (Figure 1), poisoned data may induce the 041 DNNs to erroneously learn a strong correlation between the adversary-defined trigger pattern (e.g., 042 sunglasses) and the target label (e.g., a high-authority individual). While behaving normally without 043 the trigger, the backdoored model predicts any individuals wearing sunglasses as the pre-determined 044 high-authority person. Following the current mainstream research on backdoor attacks, we focus on image classification tasks as the entry point for studying backdoor defenses.

Recently, a branch of in-training defenses has focused on training benign models directly from poisoned data, which is particularly significant when developing our own models using untrustworthy datasets. They primarily adhere to a data-splitting paradigm that differentiates between benign and poisoned samples, and disrupts the association between trigger patterns and target labels to mitigate backdoor behaviors. To name a few, Anti-backdoor learning (ABL) (Li et al., 2021a) isolates poisoned samples through local gradient ascent and unlearns the underlying malicious pattern. Decoupling-based defense (DBD) (Huang et al., 2022) utilizes self-supervised learning for a secure feature extractor to identify unconfident samples as malicious. Adaptive splitting-based defense (ASD) (Gao et al., 2023) further introduces meta-split to identify clean hard samples.



Figure 1: Illustration of data-poisoning-based backdoor attacks.

064 Intuitively, clean subsets play a vital role in various backdoor defenses (Zhu et al., 2024; Liu et al., 065 2018; Wu & Wang, 2021; Zeng et al., 2021; Li et al., 2023a), as they could provide insight into 066 the decision-making features of clean samples (e.g., as long as we know the facial characteristics 067 of people, we can accurately recognize them without being misled by accessories). In-training 068 defenses rely on clean seed samples for better performance (Gao et al., 2023). However, recollecting a clean subset can be extremely expensive when the training set has a large number of classes 069 (e.g., collecting new benign facial records for millions of people within the facial database), and manually checking a large training set to select a clean subset is time-consuming and risky for privacy 071 leaks. In addition, the potential presence of stealthy malicious 'clean sets' can further undermine the 072 effectiveness of these defenses, as triggers like sunglasses and image warping are hard to identify. 073 While certain in-training defenses seek to work without clean subsets, they may involve complicated 074 feature extraction and analysis, suffering from significant training costs (Huang et al., 2022) and 075 compromised performance (Li et al., 2021a; Chen et al., 2022a; Tran et al., 2018; Weber et al., 2023; 076 Liu et al., 2023).

In this work, we focus on improving the state-of-the-art in-training defense under the challenging non-clean-seed-involved scenario. We identify the insufficient utilization of the easier-to-obtain target labels and clean hard samples of existing methods, and propose a <u>B</u>i-perspective <u>S</u>plitting <u>D</u>efense (BSD) that splits the dataset using both semantic and loss statistics characteristics of poisoned samples.

Specifically, BSD first initialize the clean and poison pools through open set recognition-based 083 splitting (OSS) and altruistic model-based data splitting (ALS). OSS reframes the identification of 084 poisoned samples within the target class as an open-set recognition problem. Non-target classes are 085 designated as known-known classes (KKCs) to warm up the main model, thus true clean samples within the target class are distinguished as an unknown-unknown class (UUC) because their semantic 087 information is unseen to the model. ALS utilizes an altruistic model to reveal reliable clean hard 880 samples with high loss difference to the main model. Since the above two mechanisms complement 089 each other by employing different judgment perspectives, the intersection of their results provides a 090 robust initialization.

Subsequently, to prevent potential underfitting of certain classes (i.e., clean pools do not encompass all the classes) and to capture evasive poisoned samples (i.e., clean pools include some poisoned samples), BSD adopts class completion and selective dropping strategies during subsequent pool updates, ameliorating the loss-perspective-only splitting result.

095 096

098

099

100

102

103

104

105

063

In summary, our main contributions are:

- We investigate the realistic and challenging task of training time backdoor defense without clean seed samples, and identify two main breakthrough points of the problem.
- We propose two novel pool initialization mechanisms in BSD, namely ALS and OSS. They leverage the loss statistics of clean hard samples based on altruistic models and reframe the splitting as an open set recognition task for better initialization respectively, accomplishing effective backdoor defense free from the clean seed samples.
- We introduce two new pool update strategies based on the altruistic model to address the potential collapse. Class completion and selective dropping deal with the missing classes and evaded poison samples respectively.
- Extensive experiments demonstrate that BSD has an average improvement in Defense Effectiveness Rating (DER) by 16.29% compared to 5 state-of-the-art defenses.

108 2 RELATED WORKS

¹¹⁰ Currently, the countermeasures for backdoor attacks fall into two main categories:

Post-training backdoor defenses focus on repairing a backdoored model with a set of locally prepared clean training sets. Trigger inversion (Sur et al., 2023) is a popular method to reconstruct the trigger pattern and then unlearn it to renovate the model. In addition to trigger-synthesis defenses, pruning, distillation, finetuning, and model connectivity analysis (Liu et al., 2018; Wu & Wang, 2021; Li et al., 2023a) are widely applied in the realm of backdoor defense as well. Despite the promising results, most post-training methods assume using an extra clean set for defense, which introduces potential limitations.

118 **In-training backdoor defenses** aim at training a benign model from the polluted dataset, which holds 119 considerable practical significance (Chen et al., 2022a; Tran et al., 2018; Weber et al., 2023; Liu et al., 120 2023). Following an intuitive idea of splitting the dataset into clean and poison pools and treating them 121 separately, several representative training-time defenses, namely Anti-backdoor learning (ABL) (Li 122 et al., 2021a), Decoupled-based defense (DBD) (Huang et al., 2022), and Adaptive splitting-based 123 defense (ASD) (Gao et al., 2023), have garnered attention. Anti-backdoor learning (ABL) (Li et al., 124 2021a) isolates a small ratio of poisoned samples through local gradient ascent and unlearns these 125 samples to neutralize the effect of remaining poisoned samples in the clean pool. Decoupling-based 126 defense (DBD) (Huang et al., 2022) utilizes self-supervised learning to acquire a benign feature extractor and uses a clean subset to initialize the classifier head. Then, it separates the suspicious 127 according to the loss magnitude and breaks the link between the trigger and the target label through 128 semi-supervised learning. Adaptive splitting-based defense (ASD) (Gao et al., 2023) further improves 129 the initialization based on clean seed samples and introduces meta-split to identify clean hard samples, 130 achieving higher clean accuracy (CA). Besides these defenses, adopting differential-privacy SGD (Du 131 et al., 2019) and strong data augmentation (Borgnia et al., 2021) can also defend against backdoor 132 attacks to some degree. Our BSD belongs to the data-splitting in-training defenses and makes further 133 adaptions.

134 135

136 137

3 PRELIMINARIES

137 3.1 THREAT MODEL

Following Gao et al. (2023), We adopt the poisoning-based threat model used in previous works (Gu et al., 2017; Chen et al., 2017; Turner et al., 2018), where the training dataset contains a set of pre-crafted poisoned samples provided by attackers. As a typical setting of training-time defenses in previous works (Gao et al., 2023; Borgnia et al., 2021; Du et al., 2019; Huang et al., 2022; Li et al., 2021a), we assume that defenders have control over the training process.

144 145

3.2 PROBLEM FORMULATION

146 The malicious training set from the adversaries can be denoted as $\mathcal{D} = \mathcal{D}_c \cup \mathcal{D}_p$, where \mathcal{D}_c is a subset 147 of the raw benign dataset $\mathcal{D}_{raw} = \{(x_i, y_i)\}_{i=1}^N$. Each $x_i \in \mathcal{X} \subset \mathbb{R}^{C \times W \times H}$. The ground-truth 148 labels $y_i \in \mathcal{Y} = \{0, 1, \dots, C-1\}$, with C being the number of categories. Given the poisoning rate 149 ρ, \mathcal{D}_c has $(1-\rho)N$ samples. The poisoned set $\mathcal{D}_p = \{(G(x), T(y)) \mid (x, y) \in \mathcal{D}_{raw} \setminus \mathcal{D}_c\}$, where 150 $G: \mathcal{X} \to \mathcal{X}, T: \mathcal{Y} \to \mathcal{Y}$ are the attack-specific poisoned image generator and label modifier. As an 151 example, $G(x) = m \odot x + (1 - m) \odot t$, $T(x) = y_t$, where the mask $m \in \{0, 1\}^{C \times W \times H}$, $t \in \mathcal{X}$ is 152 the trigger pattern, and y_t is the target label. We call the $\{(x, y) | y \neq y_t, (x, y) \in \mathcal{D}\}$ as non-target 153 samples \mathcal{D}_{nt} , and $\{(x,y)|y=y_t, (x,y)\in \mathcal{D}\}$ as target samples $\mathcal{D}_t, \{(x,y)|y=y_t, (x,y)\in \mathcal{D}_b\}$ as 154 clean target samples \mathcal{D}_{ct} . 155

Following the natural idea to exclude the poison samples from the training set, defenders can divide D into a clean pool $\mathcal{D}_{\tilde{c}}$ and a poison pool $\mathcal{D}_{\tilde{p}}$. To prevent the model from being backdoored while preserving the performance on benign samples, the core is breaking the link between triggers and target labels, and making the best of the poison pool. We follow DBD and ASD to use semi-supervised learning (Berthelot et al., 2019b) that only leverages visual features of samples in the poison pool:

161
$$\mathcal{L}_{\text{semi}} = \sum_{(x,y)\in\mathcal{D}_{\bar{c}}} \mathcal{L}_s(x,y;\theta) + \lambda \sum_{x\in\mathcal{D}_{\bar{c}}\setminus\mathcal{D}_{\bar{p}}} \mathcal{L}_u(x;\theta), \qquad (1)$$



Figure 2: An overview of our BSD. BSD consists of two main initialize mechanisms, i.e., open set recognition-based splitting (OSS) and altruistic model-based splitting (ALS). BSD use the intersection of $\mathcal{D}_{c_{oss}}$ and $\mathcal{D}_{c_{als}}$ from OSS and ALS as the clean pool initialization. In the next two stages of the subsequent training, BSD dynamically updates the clean and poison pools based on a loss-guided split strategy based on the loss discrepancy of the main model f_{θ} and the altruistic model g_{φ} . The pseudo-code of BSD is provided in Appendix B.

185 186 187

195

196 197

199 200

201

202

203

204

181

182

183

where θ denote the weights of the main model $f(x; \theta)$ (f_{θ} for simplicity), \mathcal{L}_s is a common supervised loss function such as cross-entropy loss, the unsupervised \mathcal{L}_u is applied on the suspicious polluted set $\mathcal{D}_{\tilde{c}} \setminus \mathcal{D}_{\tilde{p}}$, with a trade-off coefficient λ . Appendix C.6 provides a detailed definition of semi-supervised learning.

The main task under this framework lies in finding an appropriate indicator that helps maximize the difference between benign and poisoned samples, thus returning a clean pool with high precision and a poison pool with high recall, i.e.:

$$\min_{\mathcal{D}_{\tilde{c}}} |\mathcal{D}_p \cap \mathcal{D}_{\tilde{c}}| \text{ s.t. } \mathcal{D}_{\tilde{c}} \subset \mathcal{D}, \ \max_{\mathcal{D}_{\tilde{c}}} |\mathcal{D}_p \cap \mathcal{D}_{\tilde{p}}| \text{ s.t. } \mathcal{D}_{\tilde{p}} \subset \mathcal{D}.$$
(2)

4 PROPOSED METHOD

Our BSD has three main components as illustrated in Figure 2. As we assume no extra clean subset access, pool initialization is vital to the defense. To ensure a secure initialization, open set recognition-based splitting (OSS) and altruistic model-based splitting (ALS) focus on the perspectives of image **semantic information** and **loss statistics** respectively. Based on the altruistic model introduced in ALS, we further improve the pool update with class completion and selective dropping strategy.

(1) **OSS** is motivated by the similarity between the open-set recognition task and poison sample detection in backdoor defense. As the main model is warmed up using \mathcal{D}_{nt} , poison samples are unknown-known-classes (UKCs) whose semantic information is included in \mathcal{D}_{nt} , thus having smaller minimum distances to feature clusters of known-known-classes (KKCs). Clean target samples fall into a new cluster and have larger minimum distances. Detailed description in Section 4.1.1.

(2) ALS highlights the clean hard samples with high loss values in the altruistic model, which could filter out the overfitted poison samples. A detailed description of ALS is provided in Section 4.1.2.

(3) Subsequent training of BSD follow a loss-guided split, which uses the loss difference of a sample
 between the main and altruistic model to distinguish samples. BSD compensates the less selected
 categories and drops the evaded poison samples using class completion and selective dropping
 strategies respectively. A detailed description of subsequent training is provided in Section 4.2.

4.1 The initialization of clean and poison pools

The initialization of the clean and poison pool is then obtained by intersecting the consensual clean samples in ALS and OSS:

$$\mathcal{D}_{\tilde{c}} = \mathcal{D}_{c_{als}} \cap \mathcal{D}_{c_{oss}}, \ \mathcal{D}_{\tilde{p}} = \mathcal{D}_{p_{als}} \cup \mathcal{D}_{p_{oss}}, \tag{3}$$

where the $\mathcal{D}_{c_{als}}$ and $\mathcal{D}_{p_{als}}$ is the split result of ALS, $\mathcal{D}_{c_{oss}}$ and $\mathcal{D}_{p_{oss}}$ is the split result of OSS. The following two subsections will explain the two initialization mechanisms.

4.1.1 OPENSET RECOGNITION BASED SPLITING

220 221 222

223

224 225

226

247

248 249

250

251

252 253 254

255

256

257 258

259 260

Openset recognition (OSR) is a task that aims to identify whether a test sample belongs to one of the semantic classes in a classifier's training set. In the context of OSR, identifying unknown-known classes (UKCs) and unknown-unknown classes (UUCs) are two major tasks. Here UKCs refer to classes for which no samples are available in training, but their side information (such as semantic/attribute information, etc.) can be obtained during training. UUCs refer to classes that do not have any relevant information during the training process: not only have they not been seen, but there is also no side information during the training process.

We notice that distinguishing the clean target samples and poison samples is related to the UKCs and UUCs identification in OSR. The poison samples are sort of UKCs because the triggers do not corrupt their semantic information. Hence, we set out to cast the clean target samples to UUCs, which can reframe the splitting within the target class into an OSR problem.

To make the poison samples and clean target samples belong to the UKCs and UUCs respectively, the known-known classes (KKCs, i.e. the training set) should contain the semantic classes of UKCs (D_p), while information of the UUCs (D_{ct}) is not included. Therefore, we construct the KKCs with the non-target classes (D_{nt}) which satisfies both requirements above. Thus, we can train the main model f_{θ} on D_{nt} for its warm-up, i.e., $\theta = \operatorname{argmin} \mathcal{L}_{semi}(D_{nt}; f_{\theta})$.

Now the local detection of poisoned samples in \mathcal{D}_t has been reframed as an open set recognition problem. The clean pool identified by OSS can be acquired by adding the approximated UUCs $(\mathcal{D}_{\tilde{ct}})$ to the KKCs $(\mathcal{D}_{nt} = \{(x, y) | y \neq y_t, (x, y) \in \mathcal{D}\})$:

 $\mathcal{D}_{c_{\rm oss}} = \mathcal{D}_{\rm \tilde{c}t} \cup \mathcal{D}_{\rm nt}, \ \mathcal{D}_{p_{\rm oss}} = \mathcal{D} \backslash \mathcal{D}_{c_{\rm oss}}.$ (4)

To approximate the \mathcal{D}_{ct} in the reframed problem, it's ideal to have the known-unknown classes (KUCs), which again indicates the need for clean seed samples. Fortunately, there have been a lot of previous studies on solving this problem without KUCs. We approximate \mathcal{D}_{ct} by:

$$\mathcal{D}_{\tilde{\mathsf{ct}}} = \left\{ (x, y) \mid \mathcal{S}(x) \ge \text{Percentile}\left(\mathcal{D}_{\mathsf{t}}^{\mathcal{S}}, 1 - \beta\right) \right\},\tag{5}$$

where β is a fixed ratio of samples in \mathcal{D}_t to be added to \mathcal{D}_{nt} , $\mathcal{D}_t^S = \{S(x) \mid (x, y) \in \mathcal{D}_t\}$ is the mapped \mathcal{D} using S. Motivated by OpenMAX (Bendale & Boult, 2016), we take the feature distance to KKCs as a metric to measure the likelihood of a sample within \mathcal{D}_t to be a true clean sample:

$$\mathcal{S}(x) = \min_{i = \{0, 1, \dots, C-1\} \setminus \tilde{y_i}} \left\{ ||f_e(x) - \mu_i||_2 \right\},\tag{6}$$

where f_e is the feature extractor of f, $\mu_i = \frac{1}{N_i} \sum f_e(x_i)$ it the cluster center of each KKC.

262 Approximating y_t . It should be noted that it requires y_t to construct \mathcal{D}_t and \mathcal{D}_{nt} . Although the target la-263 bel y_t used in the above process is unknown to the defender, it's easy to approximate. There exist vari-264 ous alternative methods to detect the y_t (Gao et al., 2024; Zhu et al., 2024), we here adopt a lightweight 265 solution by just slightly modifying the warm-up of the altruistic model. We add local gradient as-266 cent (Li et al., 2021a) and a local voting process: $\tilde{y}_t = \arg \max_c |\{(x, y) \mid y = c \land (x, y) \in \mathcal{D}_{lqa}\}|$ where \mathcal{D}_{lqa} denotes the isolated 1% samples having the smallest loss values after local gradient ascent 267 268 training on the altruistic model. In common scenarios where the dataset is a large but well-known benchmark dataset, the number of samples in each class is known to the public, y_t can be just 269 approximated through label statistics. Appendix B provides a detailed description of this process.

4.1.2 ALTRUISTIC MODEL BASED SPLITTING

In our BSD, we introduce an altruistic model $g(x; \varphi)$ (g_{φ} for simplicity), which is an independent model having the same structure as the main model. It serves as a pathfinder of the main model by exposing itself to the entire malicious training set, i.e., $\varphi = \operatorname{argmin} \mathcal{L}_{ce}(\mathcal{D}, g_{\varphi})$, where \mathcal{L}_{ce} stands for the cross-entropy loss.

We calculate the rest unsolved part in equation 3, i.e., $\mathcal{D}_{c_{als}}$ and $\mathcal{D}_{p_{als}}$ following the equation below:

$$\mathcal{D}_{c_{als}} = \left\{ (x, y) \,|\, \mathcal{L}(x, y, \varphi) \geq \text{Percentile} \left(\mathcal{D}^{\mathcal{L}}, 1 - \alpha \right) \right\}, \ \mathcal{D}_{p_{als}} = \mathcal{D} \setminus \mathcal{D}_{c_{als}}, \tag{7}$$

where \mathcal{L} is the symmetric cross-entropy loss (Wang et al., 2019), $\mathcal{D}^{\mathcal{L}} = \{\mathcal{L}(x, y, \varphi) \mid (x, y) \in \mathcal{D}\}$ is loss values using g_{φ} of the training set, Percentile returns the α -percentile in $\mathcal{D}^{\mathcal{L}}$, α is the ratio of samples split to the clean pool.

Note that although here the altruistic model is just used for the pool initialization, it also plays a significant role in the subsequent training.

285286 4.2 SUBSEQUENT TRAINING

BSD adaptively updates the pools according to the loss discrepancy of f_{θ} and g_{φ} in the subsequent training, ensuring balanced and robust learning

Class completion strategy. Despite securing good pool initialization without involving the clean seed samples, the clean pools may have an unbalanced distribution of classes, hampering the model's performance on clean accuracy. This primarily stems from the imbalanced learning status of categories and the cyclic positive feedback effect of loss-guided methods. We further revise the splitting strategy of clean samples, adding samples in the class with the fewest samples:

$$\mathcal{D}_{\tilde{c_1}} = \left\{ (x, y) | \mathcal{I}(x, y) \ge \text{Percentile} \left(\mathcal{D}^{\mathcal{I}}, 1 - \alpha \right) \lor \left\{ y = i, \mathcal{I}(x, y) \ge \text{Percentile} \left(\mathcal{D}_i^{\mathcal{I}}, 1 - n_i'/N_i \right) \right\} \right\}, \\ \mathcal{D}_{\tilde{v_1}} = \mathcal{D} \setminus \mathcal{D}_{\tilde{c_1}},$$

where $\mathcal{I}(x, y)$ is an loss based indicator, $\mathcal{D}^{\mathcal{I}} = \{\mathcal{I}(x, y) \mid (x, y) \in \mathcal{D}\}$ is the mapped \mathcal{D} using \mathcal{I} . $\mathcal{D}_{i}^{\mathcal{I}} = \{\mathcal{I}(x, y) \mid y = i(x, y) \in \mathcal{D}\}, N_{i} = |\mathcal{D}_{i}^{\mathcal{I}}|, n_{i}' = \min\{\alpha n_{i}, N_{secondFew}\}, N_{secondFew}$ is number of samples in the second-fewest predicted class.

We do subtraction between the loss of samples on the main and altruistic models, as the poison samples should also have high loss values on the unaffected main model and low loss values on the backdoored altruistic model. Thus \mathcal{I} is defined as:

$$\mathcal{I}(x,y) = \mathcal{L}_{sce}(x,y,\varphi) - \mathcal{L}_{sce}(x,y,\theta), \qquad (9)$$

(8)

307 where \mathcal{L}_{sce} denotes the symmetric cross-entropy loss (Wang et al., 2019).

Selective dropping strategy. Approaching the end of the training, we drop the samples that are predicted to be \tilde{y}_t by both models:

$$\mathcal{D}_{\tilde{c}_{2}} = \mathcal{D}_{\tilde{c}_{1}} \setminus \{ (x, y) \mid (f(x) = \tilde{y}_{t}) \land (g(x) = \tilde{y}_{t}) \}, \ \mathcal{D}_{\tilde{p}_{2}} = \mathcal{D} \setminus \mathcal{D}_{\tilde{c}_{2}},$$
(10)

There exist two probable situations for a sample that will be dropped: 1) the sample is poisoned; 2) the sample is a clean sample with the original label being \tilde{y}_t . For situation 1, it is the correct decision to drop poisoned samples; for situation 2, the agreement between the two models indicates the sample is already well-fitted by both models and is less important. As a result, the dropping of these samples generally helps improve model performance.

317 318

319

321

311

277 278

295 296 297

302

303

304 305 306

5 EXPERIMENTS

320 5.1 EXPERIMENTAL SETTINGS

Datasets and DNN models. We adopt three benchmark datasets for the evaluation of the backdoor defenses, namely, CIFAR-10 (Krizhevsky et al., 2009), GTSRB (Stallkamp et al., 2012), and Imagenet (Deng et al., 2009). The results are conducted with ResNet-18 (He et al., 2016) and MobileNet-v2 (Sandler et al., 2018) as the backbone models for their representativeness and
 widespread use.

Attack Baselines. We implement seven representative attacks, i.e., BadNets (Gu et al., 2017), 327 Blended (Chen et al., 2017), WaNet (Nguyen & Tran, 2021), Label-Consistent(LC) (Turner et al., 328 2019), ReFool (Liu et al., 2020), SIG (Barni et al., 2019), and Narcissus (Zeng et al., 2023b). All these attacks are implemented based on open-source codebases of ASD (Gao et al., 2023), DBD (Huang 330 et al., 2022), Narcissus (Zeng et al., 2023b), backdoorBench (Wu et al., 2022), and BackdoorBox (Li 331 et al., 2023b). The first five attacks follow the same setting in settings in (Gao et al., 2023) unless 332 otherwise specified, SIG and Narcissus follow the setting with Li et al. (2021a) and Zeng et al. 333 (2023b) respectively, while the poisoning rate ρ and target label y_t are the same as LC. A detailed 334 description of the attack implementations is provided in Appendix C.3.

335 Defense Baselines. We compare our proposed BSD with five existing backdoor defenses, namely Fine-336 pruning (FP) (Liu et al., 2018), Neural Attention Distillation (NAD) (Li et al., 2021b), Anti-Backdoor-337 Learning (ABL) (Li et al., 2021a), Decoupling-based Backdoor Defense (DBD) (Huang et al., 2022), 338 and Adaptive Splitting-based backdoor Defense (ASD) (Gao et al., 2023). The detailed settings for 339 all defense baselines are as suggested in ASD. For our BSD, we adopt the MixMatch (Berthelot 340 et al., 2019b) semi-supervised training framework for the main model, following Decoupling-based 341 Defense (DBD) and Adaptive Splitting-based Defense (ASD). The altruistic model undergoes a warm-up phase with 25 epochs, utilizing the Adam optimizer, cross entropy loss, with a learning rate 342 of 0.001. The default warm-up epochs for the main model in OSS are set to 20 ($T_1 = 20$), with a 343 default fixed β of 0.2. Class completion training spans 60 epochs ($T_2 = 90$), and selective dropping 344 training spans 30 epochs ($T_3 = 120$). The clean pool ratio α follows a sinusoidal growth curve 345 during class rebalance training, starts at 0.2, and reaches an upper limit of 0.6 at the end of the class 346 completion stage, after which it remains fixed. Additional details are available in Appendix C.4. 347

Evaluation metrics. We assess the effectiveness of backdoor defenses using two widely used metrics:
Clean Accuracy (CA) and the attack success rate (ASR). To be specific, the CA is the accuracy of
clean data, the ASR is defined as the proportion of poisoned samples that are misclassified as the
target class by the model. In the context of backdoor defense, superior performance is characterized
by higher CA and lower ASR. To comprehensively evaluate the performance of defense methods, we
include another metric named Defense Effectiveness Rating (DER) (Zhu et al., 2023a), higher DER
indicate better defense performance. The detailed definition of DER is provided in Appendix C.5.

355 356

357 358

5.2 MAIN RESULTS

359 We present a summary of CAs, ASRs, and DERs achieved by five backdoor defenses against three 360 most representative backdoor attacks on three benchmark datasets in Table 1^1 . As illustrated in Table 1, our BSD has the best average DERs on each dataset, being capable of maintaining high CAs 361 without compromising the robustness indicated by ASRs. In comparison with post-training defenses, 362 i.e., FP and NAD, which require thousands of clean seed samples, BSD consistently outperforms 363 them with lower ASRs when OSS is used as the alternative initialization. Additionally, the CAs of 364 BSD surpass those of FP and NAD. Concerning recently proposed training-time defenses, the BSD has best result in general. ABL, which assumes no presence of clean subsets, has relatively close 366 performance under CIFAR-10 & BadNets, GTSRB & BadNets, and GTSRB & Blend. Nevertheless, 367 the CA under CIFAR-10 & WaNet indicates a class underfitting collapse (CAs on certain classes are 368 close to 0%) and its performance is inferior to that of BSD in general. For another representative 369 training-time defense DBD, although it has a slight edge in ASRs on CIFAR-10, its average ASRs 370 and CAs fall behind our BSD. ASD, which assumes an extra small clean seed set is characterized by 371 consistent high CAs and stable ASRs. However, BSD still surpasses it in general. In summary, our 372 BSD performance remains competitive and, in some cases, surpasses that of state-of-the-art methods.

374

³⁷² 373

 ¹Since we strictly follow the same settings, we reference the baseline results for CIFAR-10 and GTSRB from
 ASD (Gao et al., 2023). However, the exact 30 randomly selected classes from the Imagenet subset used are unknown to us, so we ran all the baselines on Imagenet using our own randomly chosen 30 classes.

DATASET	ATTACK	METRIC	Non	FP	NAD	ABL	DBD	ASD	BSD(OURS)
		CA	94.9	<u>93.9</u>	88.2	93.8	92.3	93.4	95.1
	BADNET	ASR DER	100.0	1.8 98.6	4.6 94.4	1.1 98.9	0.8 98 3	1.2 98 7	<u>0.9</u> 99.6
	¦	CA	94.1	92.9	85.8	91.9	91.7	93.7	94.9
CIELD 10	BLENDED	ASR	98.3	77.1	3.4	1.6	0.7	$\frac{33.7}{1.6}$	0.8
CIFAR-10		DER	-	60.0	93.3	97.3	97.6	<u>98.2</u>	98.8
	NV. N	CA	93.6	90.4	71.3	84.1	91.4	$\frac{93.1}{1.7}$	94.5
	WANET	DER	- 99.9	98.6 49.1	6.7 85.5	2.2 94.1	0.0 98.9	1.7 98.9	<u>0.8</u> 99.6
	AVERAG	E DER	-	69.2	91.0	96.8	98.3	98.6	99.3
	1	CA	97.6	84.2	97.1	97.1	91.4	96.7	97.6
	BADNET	ASR	100.0	0.0	0.2	0.0	0.0	0.0	0.0
	I	DER	-	93.3	99.7	<u>99.8</u>	96.9	99.6	100.0
	BLENDED	CA	97.2	91.4 68.1	93.3 62.4	97.1 0.5	91.5	97.1 0.3	<u>96.9</u>
GTSRB	DLENDED	DER	-	62.8	66.6	99.4	46.9	<u>99.5</u>	99.6
	WANET	CA	97.2	92.5	96.5	97.0	89.6	97.2	97.2
		ASR	100.0	21.4	47.1	0.4	0.0	0.3	$\frac{0.2}{0.0}$
		DEK	-	87.0	/0.1	99.7	96.2	99.9	99.9
	AVERAG	JE DEK	-	81.0	80.8	99.6	80.0	99.6	99.8
	BADNET	CA ASR	75.7 99.5	71.4	51.7 2.5	68.1 7.6	76.1	81.1 100.0	$\frac{78.3}{1.1}$
	BADIGET	DER	-	<u>96.3</u>	86.5	92.2	<u>99.2</u>	50.0	99.2
		CA	74.5	73.1	42.8	61.9	77.9	<u>79.7</u>	80.1
IMAGENET	BLENDED	ASR	97.7	81.9 57.2	0.2 82.0	100.0	$\frac{35.0}{81.4}$	51.0 73.4	0.2
	¦	OLK		76.0	74.0	74.0	77.0	79.4	78.7
	WANET	ASR	77.1 81.0	0.4	1.3	74.9 1.1	5.2	$\frac{78.4}{14.0}$	0.0
		DER	-	90.2	88.3	88.9	87.9	83.5	90.5
	AVERAG	GE DER	-	81.2	85.9	74.5	89.5	69.0	96.2

378	Table 1: The clean accuracy (CA%), attack success rate (ASR%), and defense effective rating
379	(DER%) of 5 baseline backdoor defense methods and our BSD against 3 representative backdoor
380	attacks on 3 benchmark datasets. The baselines consist of two post-training defenses (FP, NAD) and
381	three state-of-the-art training-time defenses (ABL, DBD, ASD). 'Non' stands for no defense. The
382	best and second best results are in bold and underlined.

5.3 **RESISTANCE TO MORE ATTACKS**

In addition to the representative attacks presented in the main results, we investigated four more 410 attacks that may be threatening to existing defenses. They consist of one invisible attack, ReFool (Liu 411 et al., 2020), and three clean-label attacks, LC (Turner et al., 2019), SIG (Barni et al., 2019), and 412 Narcissus (Zeng et al., 2023b). ReFool uses a physical yet stealthy reflection trigger, which makes 413 the backdoor hard to detect. LC, SIG, and Narcissus belong to the clean-label attack, which is a type 414 of tricky backdoor attack that does not change the label of samples, making most of the defenses 415 ineffective (where DBD has the most significant performance degradation). For our BSD, clean-label 416 attacks are less threatening. While the OSS mechanism can be evaded as the semantic information is 417 \mathcal{D}_{t} is consistent. Fortunately, ALS still functions effectively with its loss-perspective splitting in this 418 scenario, compensating for the limitations of OSS. As shown in Table 2, BSD is not evaded by any 419 of the attacks and achieves the best average DER. Additional details of attack implementation are 420 available in Appendix C.3.

421 422

423

408

409

5.4 ROBUSTNESS TO DIFFERENT MODEL STRUCTURES

BSD makes no assumptions about model structures, ensuring both compatibility and versatility. To
validate this, we evaluated the defense performance of BSD using another widely adopted network,
MobileNet (Sandler et al., 2018). As shown in Table 3, BSD consistently outperforms the baseline
method with MobileNet-v2 as the backbone.

428 429

430

5.5 ROBUSTNESS TO DIFFERENT POISONING RATES

431 Despite the default poisoning rate $\rho = 0.05$ being a reasonable setting that is widely adopted in either backdoor attack or backdoor defense research (Huang et al., 2022; Gao et al., 2023; Min et al.,

ATTACK	METRIC	Non	FP	NAD	ABL	DBD	ASD	BSD(OURS)
LC	CA ASR DER	94.4 99.9 -	87.1 24.4 84.1	85.9 50.5 70.5	80.2 <u>1.6</u> <u>92.1</u>	83.2 98.1 45.3	93.9 73.2 63.1	<u>92.4</u> 1.2 98.4
SIG	CA ASR DER	95.0 95.2	87.1 60.8 63.3	85.8 83.0 51.5	67.6 5.1 81.4	80.1 99.9 42.6	$\frac{93.5}{96.5}$ 49.3	93.8 0.0 97.0
ReFool	CA ASR DER	95.2 99.0	86.5 23.0 83.6	85.6 42.5 73.4	76.3 82.0 49.0	90.8 2.3 <u>96.1</u>	86.8 0.4 95.1	94.8 <u>0.5</u> 99.0
NARCISSUS	CA ASR DER	95.2 99.5	87.2 63.4 64.0	86.5 81.0 54.8	79.3 7.1 88.2	87.3 99.5 46.0	<u>93.9</u> 0.0 <u>99.1</u>	94.3 0.0 99.3
AVERAGE DER		-	73.8	62.6	77.7	57.5	<u>76.6</u>	98.4

Table 2: The clean accuracy (CA%), attack success rate (ASR%), and defense effective rating (DER%) of 5 baseline backdoor defense methods and our BSD against 4 threatening backdoor attacks on CIFAR-10. The best and second best results are in **bold** and underlined.

Table 3: The clean accuracy (CA%), attack success rate (ASR%), and defense effectiveness rating (DER%) on CIFAR-10 of different defenses using mobilenet v2 (Sandler et al., 2018) as the backbone.

ATTACK	METRIC	Non	FP	NAD	ABL	DBD	ASD	BSD(OURS)
BADNET	CA ASR DER	94.3 100.0	77.9 8.3 <u>87.7</u>	78.5 11.7 86.2	79.7 13.6 85.9	65.5 0.0 85.6	93.2 100.0 49.4	$\frac{91.1}{0.4}$ 98.2
Blended	CA ASR DER	94.0 99.3	75.9 30.8 75.2	76.0 46.0 67.7	67.3 2.6 85.1	69.0 0.0 <u>87.2</u>	$\frac{87.1}{99.0}$ 46.7	90.0 <u>0.2</u> 97.6
WANET	CA ASR DER	94.0 95.7	82.2 2.4 <u>90.7</u>	81.5 3.2 90.0	50.9 0.5 76.1	58.4 12.4 73.9	$\frac{83.0}{97.7}$ 44.5	90.1 <u>0.6</u> 95.6
AVERA	GE DER	-	84.5	81.3	82.3	82.2	46.9	97.1



Figure 3: The performance of BSD in comparison with ASD (Gao et al., 2023) under different poisoning rates. The experiment is conducted on CIFAR-10 against three attacks.

2024; Shi et al., 2023), it's crucial to verify the robustness of our BSD under different poisoning rates. As illustrated in Figure 3, although ASD performs well with respect to ASRs as well, the CAs of ASD are conspicuously lower under non-default settings. However, our BSD consistently achieves close-to-zero ASRs and satisfying CAs, emphasizing its robustness to different poisoning rates.

5.6 TRAINING COST

Our BSD incorporates an altruistic model to assist with pool initialization and updates, which may raise concerns about increased training costs. However, as shown in Table 4, the training cost of BSD is comparable to, or even lower than, that of ASD (Gao et al., 2023). This is due to three key factors: 1) The altruistic model is updated through standard training rather than MixMatch, significantly reducing time. 2) The altruistic model is only updated before stage 3, and its training primarily runs in parallel with the main model. 3) An imbalanced pool size, as seen in the early stages of ASD, often triggers frequent dataloader updates in MixMatch, whereas the clean pool size in BSD is more balanced and suitable during training.

Method	CIFAR-10	GTSRB	IMAGENET	AVERAGE
DBD	11.96	10.09	53.21	25.09
ASD	4.81	2.55	12.09	6.48
BSD(OURS)	3.15	2.84	9.20	5.06

Table 4: Training cost (hours) of ASD, DBD, and BSD on CIFAR-10, GTSRB, and Imagenet.

5.7 ABLATION STUDIES

Effectiveness of different stages. The major components of BSD are divided into pool initialization and pool updates. We investigated the significance of each component on CIFAR-10 to demonstrate their necessity, as shown in Table 5. OSS and ALS initialization are critical for avoiding backdoor overfitting (ASR); class completion update helps prevent class underfitting (CA); and selective dropping update acts as a final step to further reduce ASR, thereby achieving a higher DER.

Table 5: The ablation study on the strategies involved in BSD under CIFAR-10. 'Default' represents the result using all the proposed mechanisms, 'w/o Init' represents the results using random initialization. 'w/o Completion' represents disabling class completion in both stages 2 and 3. 'w/o Drop' represents disabling selective drop in stage 3.

Setting↓	CA	BADNET ASR	DER	CA	Blended ASR	DER	CA	WANET ASR	DER
DEFAULT	95.1	$0.9 \\ 100.0 \\ 0.0 \\ 1.1$	99.6	94.9	0.8	98.8	94.5	0.8	99.6
W/O INIT	94.7		49.9	95.0	99.2	49.6	93.6	91.5	54.2
W/O COMPLETION	90.7		97.9	86.8	0	95.5	89.8	0.2	98.0
W/O DROP	94.6		99.3	94.2	1.1	98.6	94.5	1.9	99.0

Influence of parameters. We here present the influence of the main parameter, i.e., the parameters α & β for pool size control. As revealed in Table 6, the performance is good near the default setting, while an extreme setting will lead to degradation on DER.

Table 6: Performance of BSD under different $\alpha \& \beta$ on CIFAR-10. The results that have more than 0.5% DER decrease are marked using \downarrow .

515 516	Setting↓	СА	BADNET ASR	DER	CA	BLENDED ASR	DER	CA	WANET ASR	DER
517	DEFAULT	95.1	0.9	99.6	94.9	0.8	98.8	94.5	0.8	99.6
518 519 520 521	$\begin{array}{c} \alpha = 0.3, \ \beta = 0.2 \\ \alpha = 0.4, \ \beta = 0.2 \\ \alpha = 0.5, \ \beta = 0.2 \\ \alpha = 0.7, \ \beta = 0.2 \\ \alpha = 0.8, \ \beta = 0.2 \\ \alpha = 0.9, \ \beta = 0.2 \end{array}$	94.9 95.0 95.0 95.2 95.0 93.6	1.2 1.2 1.7 0.8 1.1 0.7	99.4 99.4 99.2 99.6 99.5 99.0↓	94.8 94.7 94.2 95.1 92.8 90.3	0.6 0.9 0.8 0.5 0.5 0.2	98.9 98.7 98.7 98.9 98.2↓ 97.2↓	94.2 94.4 94.7 93.0 91.7 90.0	1.8 1.4 1.4 0.1 1.0 0.9	99.1↓ 99.3 99.2 99.6 98.5↓ 97.7↓
22 23 24	$\begin{array}{c} \alpha = 0.6, \ \beta = 0.1 \\ \alpha = 0.6, \ \beta = 0.3 \\ \alpha = 0.6, \ \beta = 0.5 \\ \alpha = 0.7, \ \beta = 0.7 \end{array}$	94.8 94.9 94.7 94.4	0.7 1.2 1.4 1.7	99.6 99.4 99.2 98.9↓	90.9 94.7 94.9 94.3	0.5 0.8 1.6 3.7	97.3↓ 98.8 98.4 97.3↓	91.6 94.5 94.3 94.0	0.6 0.8 2.3 32.8	98.7↓ 99.6 98.8↓ 83.6↓

5.8 OTHER EXPERIMENTS

Additional experimental results, including visualizations, extended ablation studies, potential adaptive attacks, performance in the absence of attacks, the OSS distance metric, and more, are provided in Appendix E.

CONCLUSION

In conclusion, our proposed BSD effectively mitigates backdoor attacks through bi-perspective splitting mechanisms, without relying on on extra clean data. By leveraging OSS and ALS for robust dataset splitting, combined with class completion and selective dropping strategies, BSD achieves superior backdoor defense performance. Extensive experiments confirm BSD's robustness under different attack/defense settings.

540 REFERENCES E 4 4

34 I	
542	Jean-Baptiste Alayrac, Jonathan Uesato, Po-Sen Huang, Alhussein Fawzi, Robert Stanforth, and
543	Pushmeet Kohli. Are labels required for improving adversarial robustness? Advances in Neural
544	Information Processing Systems, 32, 2019.
545	Mauro Barni, Kassem Kallas, and Benedetta Tondi. A new backdoor attack in cnns by training set

- 546 corruption without label poisoning. In 2019 IEEE International Conference on Image Processing 547 (*ICIP*), pp. 101–105. IEEE, 2019. 548
- Abhijit Bendale and Terrance E Boult. Towards open set deep networks. In Proceedings of the IEEE 549 conference on computer vision and pattern recognition, pp. 1563–1572, 2016. 550
- David Berthelot, Nicholas Carlini, Ekin D Cubuk, Alex Kurakin, Kihyuk Sohn, Han Zhang, and 551 Colin Raffel. Remixmatch: Semi-supervised learning with distribution matching and augmentation 552 anchoring. In International Conference on Learning Representations, 2019a. 553
- 554 David Berthelot, Nicholas Carlini, Ian Goodfellow, Nicolas Papernot, Avital Oliver, and Colin A 555 Raffel. Mixmatch: A holistic approach to semi-supervised learning. Advances in neural information 556 processing systems, 32, 2019b.
- Eitan Borgnia, Valeriia Cherepanova, Liam Fowl, Amin Ghiasi, Jonas Geiping, Micah Goldblum, 558 Tom Goldstein, and Arjun Gupta. Strong data augmentation sanitizes poisoning and backdoor 559 attacks without an accuracy tradeoff. In ICASSP 2021-2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pp. 3855–3859. IEEE, 2021. 561
- Nicholas Carlini and David Wagner. Towards evaluating the robustness of neural networks. In 2017 562 *ieee symposium on security and privacy (sp)*, pp. 39–57. Ieee, 2017. 563
- 564 Yair Carmon, Aditi Raghunathan, Ludwig Schmidt, John C Duchi, and Percy S Liang. Unlabeled 565 data improves adversarial robustness. Advances in neural information processing systems, 32, 566 2019. 567
- Weixin Chen, Baoyuan Wu, and Haoqian Wang. Effective backdoor defense by exploiting sensitivity 568 of poisoned samples. Advances in Neural Information Processing Systems, 35:9727–9737, 2022a. 569
- 570 Weixin Chen, Baoyuan Wu, and Haoqian Wang. Effective backdoor defense by exploiting sensitivity 571 of poisoned samples. Advances in Neural Information Processing Systems, 35:9727–9737, 2022b.
- 572 Xinyun Chen, Chang Liu, Bo Li, Kimberly Lu, and Dawn Song. Targeted backdoor attacks on deep 573 learning systems using data poisoning. arXiv preprint arXiv:1712.05526, 2017. 574
- 575 Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale 576 hierarchical image database. In 2009 IEEE conference on computer vision and pattern recognition, pp. 248-255. Ieee, 2009. 577
- 578 Min Du, Ruoxi Jia, and Dawn Song. Robust anomaly detection and backdoor attack detection via differential privacy. arXiv preprint arXiv:1911.07116, 2019. 580
- Mark Everingham, SM Ali Eslami, Luc Van Gool, Christopher KI Williams, John Winn, and Andrew 581 Zisserman. The pascal visual object classes challenge: A retrospective. International journal of 582 computer vision, 111:98-136, 2015. 583
- 584 Kuofeng Gao, Yang Bai, Jindong Gu, Yong Yang, and Shu-Tao Xia. Backdoor defense via adaptively 585 splitting poisoned dataset. In Proceedings of the IEEE/CVF Conference on Computer Vision and 586 Pattern Recognition, pp. 4005-4014, 2023.
- Yudong Gao, Honglong Chen, Peng Sun, Zhe Li, Junjian Li, and Huajie Shao. Energy-based backdoor 588 defense without task-specific samples and model retraining. In Forty-first International Conference on Machine Learning, 2024. 590
- Micah Goldblum, Dimitris Tsipras, Chulin Xie, Xinyun Chen, Avi Schwarzschild, Dawn Song, Aleksander Madry, Bo Li, and Tom Goldstein. Dataset security for machine learning: Data 592 poisoning, backdoor attacks, and defenses. IEEE Transactions on Pattern Analysis and Machine Intelligence, 45(2):1563-1580, 2022.

594 595	Tianyu Gu, Brendan Dolan-Gavitt, and Siddharth Garg. Badnets: Identifying vulnerabilities in the machine learning model supply chain. <i>arXiv preprint arXiv:1708.06733</i> , 2017.
590 597 598	Jiyang Guan, Jian Liang, and Ran He. Backdoor defense via test-time detecting and repairing. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp.
599	24564–24573, 2024.
600	Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image
601	recognition. In <i>Proceedings of the IEEE conference on computer vision and pattern recognition</i> ,
602	pp. //0–//8, 2016.
604	Shengshan Hu, Ziqi Zhou, Yechao Zhang, Leo Yu Zhang, Yifeng Zheng, Yuanyuan He, and Hai Jin.
605	Badhash: Invisible backdoor attacks against deep hashing with clean label. In <i>Proceedings of the</i> 30th ACM international conference on Multimedia, pp. 678–686, 2022.
607 608	Binxiao Huang, Jason Chun Lok, Chang Liu, and Ngai Wong. Poisoning-based backdoor attacks for arbitrary target label with positive triggers. <i>arXiv preprint arXiv:2405.05573</i> , 2024.
609 610 611	Kunzhe Huang, Yiming Li, Baoyuan Wu, Zhan Qin, and Kui Ren. Backdoor defense via decoupling the training process. <i>arXiv preprint arXiv:2202.03423</i> , 2022.
612 613 614	Andrew Ilyas, Logan Engstrom, Anish Athalye, and Jessy Lin. Black-box adversarial attacks with limited queries and information. In <i>International conference on machine learning</i> , pp. 2137–2146. PMLR, 2018.
615	Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009.
616	
617 618	In Artificial intelligence safety and security, pp. 99–112. Chapman and Hall/CRC, 2018.
619 620 621	Shaofeng Li, Minhui Xue, Benjamin Zi Hao Zhao, Haojin Zhu, and Xinpeng Zhang. Invisible backdoor attacks on deep neural networks via steganography and regularization. <i>IEEE Transactions on Dependable and Secure Computing</i> , 18(5):2088–2105, 2020.
622 623 624 625	Yige Li, Xixiang Lyu, Nodens Koren, Lingjuan Lyu, Bo Li, and Xingjun Ma. Anti-backdoor learning: Training clean models on poisoned data. <i>Advances in Neural Information Processing Systems</i> , 34: 14900–14912, 2021a.
626 627 628	Yige Li, Xixiang Lyu, Nodens Koren, Lingjuan Lyu, Bo Li, and Xingjun Ma. Neural attention distil- lation: Erasing backdoor triggers from deep neural networks. arXiv preprint arXiv:2101.05930, 2021b.
629 630 631	Yige Li, Xixiang Lyu, Xingjun Ma, Nodens Koren, Lingjuan Lyu, Bo Li, and Yu-Gang Jiang. Reconstructive neuron pruning for backdoor defense. <i>arXiv preprint arXiv:2305.14876</i> , 2023a.
632 633	Yiming Li, Yong Jiang, Zhifeng Li, and Shu-Tao Xia. Backdoor learning: A survey. <i>IEEE Transac-</i> <i>tions on Neural Networks and Learning Systems</i> , 2022.
634 635	Yiming Li, Mengxi Ya, Yang Bai, Yong Jiang, and Shu-Tao Xia. BackdoorBox: A python toolbox for backdoor learning. In <i>ICLR Workshop</i> , 2023b.
637 638 639	Kang Liu, Brendan Dolan-Gavitt, and Siddharth Garg. Fine-pruning: Defending against backdooring attacks on deep neural networks. In <i>International symposium on research in attacks, intrusions, and defenses</i> , pp. 273–294. Springer, 2018.
640 641 642	Min Liu, Alberto Sangiovanni-Vincentelli, and Xiangyu Yue. Beating backdoor attack at its own game. In <i>Proceedings of the IEEE/CVF International Conference on Computer Vision</i> , pp. 4620–4629, 2023.
643 644 645 646	Yunfei Liu, Xingjun Ma, James Bailey, and Feng Lu. Reflection backdoor: A natural backdoor attack on deep neural networks. In <i>Computer Vision–ECCV 2020: 16th European Conference, Glasgow,</i> <i>UK, August 23–28, 2020, Proceedings, Part X 16</i> , pp. 182–199. Springer, 2020.
0.47	Rui Min Zevu Oin Li Shen and Minhao Cheng. Towards stable backdoor purification through

648 649 650	Seyed-Mohsen Moosavi-Dezfooli, Alhussein Fawzi, and Pascal Frossard. Deepfool: a simple and accurate method to fool deep neural networks. In <i>Proceedings of the IEEE conference on computer vision and pattern recognition</i> , pp. 2574–2582, 2016.
651 652	Anh Nguyen and Anh Tran. Wanet–imperceptible warping-based backdoor attack. <i>arXiv preprint</i>
653 654	arXiv:2102.10369, 2021.
655 656 657	Minzhou Pan, Yi Zeng, Lingjuan Lyu, Xue Lin, and Ruoxi Jia. {ASSE1}: Robust backdoor data detection across a multiplicity of deep learning paradigms. In <i>32nd USENIX Security Symposium (USENIX Security 23)</i> , pp. 2725–2742, 2023.
658 659 660	Fanchao Qi, Mukai Li, Yangyi Chen, Zhengyan Zhang, Zhiyuan Liu, Yasheng Wang, and Maosong Sun. Hidden killer: Invisible textual backdoor attacks with syntactic trigger. arXiv preprint arXiv:2105.12400, 2021.
661 662 663 664	Mark Sandler, Andrew Howard, Menglong Zhu, Andrey Zhmoginov, and Liang-Chieh Chen. Mo- bilenetv2: Inverted residuals and linear bottlenecks. In <i>Proceedings of the IEEE conference on</i> <i>computer vision and pattern recognition</i> , pp. 4510–4520, 2018.
665 666 667	Ali Shafahi, W Ronny Huang, Mahyar Najibi, Octavian Suciu, Christoph Studer, Tudor Dumitras, and Tom Goldstein. Poison frogs! targeted clean-label poisoning attacks on neural networks. <i>Advances in neural information processing systems</i> , 31, 2018.
668 669 670 671	Yucheng Shi, Mengnan Du, Xuansheng Wu, Zihan Guan, Jin Sun, and Ninghao Liu. Black-box backdoor defense via zero-shot image purification. <i>Advances in Neural Information Processing Systems</i> , 36:57336–57366, 2023.
672 673 674 675	Kihyuk Sohn, David Berthelot, Nicholas Carlini, Zizhao Zhang, Han Zhang, Colin A Raffel, Ekin Do- gus Cubuk, Alexey Kurakin, and Chun-Liang Li. Fixmatch: Simplifying semi-supervised learning with consistency and confidence. <i>Advances in neural information processing systems</i> , 33:596–608, 2020.
676 677 678 679	J. Stallkamp, M. Schlipsing, J. Salmen, and C. Igel. Man vs. computer: Benchmarking machine learning algorithms for traffic sign recognition. <i>Neural Networks</i> , (0):-, 2012. ISSN 0893-6080. doi: 10.1016/j.neunet.2012.02.016. URL http://www.sciencedirect.com/science/ article/pii/S0893608012000457.
681 682 683 684	Indranil Sur, Karan Sikka, Matthew Walmer, Kaushik Koneripalli, Anirban Roy, Xiao Lin, Ajay Di- vakaran, and Susmit Jha. Tijo: Trigger inversion with joint optimization for defending multimodal backdoored models. In <i>Proceedings of the IEEE/CVF International Conference on Computer</i> <i>Vision</i> , pp. 165–175, 2023.
685 686	Brandon Tran, Jerry Li, and Aleksander Madry. Spectral signatures in backdoor attacks. Advances in neural information processing systems, 31, 2018.
687 688	Alexander Turner, Dimitris Tsipras, and Aleksander Madry. Clean-label backdoor attacks. 2018.
689 690 691	Alexander Turner, Dimitris Tsipras, and Aleksander Madry. Label-consistent backdoor attacks. <i>arXiv</i> preprint arXiv:1912.02771, 2019.
692 693 694 695	Hongyi Wang, Kartik Sreenivasan, Shashank Rajput, Harit Vishwakarma, Saurabh Agarwal, Jy-yong Sohn, Kangwook Lee, and Dimitris Papailiopoulos. Attack of the tails: Yes, you really can backdoor federated learning. <i>Advances in Neural Information Processing Systems</i> , 33:16070–16084, 2020.
696 697 698 699	Yisen Wang, Xingjun Ma, Zaiyi Chen, Yuan Luo, Jinfeng Yi, and James Bailey. Symmetric cross entropy for robust learning with noisy labels. In <i>Proceedings of the IEEE/CVF international conference on computer vision</i> , pp. 322–330, 2019.
700 701	Maurice Weber, Xiaojun Xu, Bojan Karlaš, Ce Zhang, and Bo Li. Rab: Provable robustness against backdoor attacks. In 2023 IEEE Symposium on Security and Privacy (SP), pp. 1311–1328. IEEE, 2023.

=00	
702	Emily Wenger, Josephine Passananti, Arjun Nitin Bhagoji, Yuanshun Yao, Haitao Zheng, and Ben Y
703	Zhao, Backdoor attacks against deep learning systems in the physical world. In <i>Proceedings of the</i>
704	IFFF/CVF conference on computer vision and pattern recognition pp 6206–6215 2021
705	The conference on comparer vision and parern recognition, pp. 0200-0210, 2021.
706	Baovuan Wu, Hongrui Chen, Mingda Zhang, Zihao Zhu, Shaokui Wei, Danni Yuan, and Chao
700	Shen, Backdoorbench: A comprehensive benchmark of backdoor learning. Advances in Neural
707	Information Processing Systems, 35:10546–10559, 2022.
709	Dension Wu and Vison Wang. Advanced number among surface healtdoored door models.
710	Adversarian Wu and Tisen wang. Adversarian neuron prunnes backdoored deep models.
710	Advances in Neural Information Processing Systems, 54.10915–10923, 2021.
/	Oizhe Xie, Zihang Dai, Eduard Hovy, Thang Luong, and Quoc Le. Unsupervised data augmentation
712 713	for consistency training. Advances in neural information processing systems, 33:6256–6268, 2020.
71/	
714	Xiaohui Zeng, Chenxi Liu, Yu-Siang Wang, Weichao Qiu, Lingxi Xie, Yu-Wing Tai, Chi-Keung Tang,
715	and Alan L Yuille. Adversarial attacks beyond the image space. In <i>Proceedings of the IEEE/CVF</i>
716	Conference on Computer Vision and Pattern Recognition, pp. 4302–4311, 2019.
717	
718	Yi Zeng, Si Chen, Won Park, Z Morley Mao, Ming Jin, and Ruoxi Jia. Adversarial unlearning of
719	backdoors via implicit hypergradient. arXiv preprint arXiv:2110.03735, 2021.
720	\mathbf{X}^{\prime} (7.5.) \mathbf{M}^{\prime} (1.5.) \mathbf{M}^{\prime} (1.5.) \mathbf{M}^{\prime} (1.5.) \mathbf{M}^{\prime} (1.5.) \mathbf{M}^{\prime} (1.5.) \mathbf{M}^{\prime} (1.5.) \mathbf{M}^{\prime}
720	Yi Zeng, Minzhou Pan, Himanshu Jahagirdar, Ming Jin, Lingjuan Lyu, and Ruoxi Jia. {Meta-Sitt}:
721	How to sift out a clean subset in the presence of data poisoning? In 32nd USENIX Security
722	<i>Symposium (USENIX Security 23)</i> , pp. 1667–1684, 2023a.
723	Vi Zang Minghou Dan Hoong Anh Just Lingiuan Lun Maikang Oiu, and Puovi Jia. Narajaguat A
724	11 Zeng, Milzhou Fai, Hodag Anii Just, Engjuan Eyu, Metkang Qiu, and Kuoxi Jia. Naterssus. A
725	practical clean-tabel backdoor attack with nimited information. In <i>Proceedings of the 2023 ACM</i>
726	SIGSAC Conference on Computer and Communications Security, pp. 1/1–185, 2023b.
707	Zaivi Zhang, Jinguan Jia, Binghui Wang, and Nail Zhangiang Gong, Backdoor attacks to graph
720	pausi patworks. In Proceedings of the 26th ACM Supposition on Access Control Models and
/28	Tachalasia np 15 26 2021
729	<i>Technologies</i> , pp. 13–20, 2021.
730	Zaixi Zhang, Oi Liu, Zhicai Wang, Zenu Lu, and Oingyong Hu. Backdoor defense via deconfounded
731	representation learning. In Proceedings of the IEFE/CVF Conference on Computer Vision and
732	Pattern Recognition pp 12728 12738 5073
733	<i>Tunem</i> Recognition, pp. 12220–12230, 2023.
72/	Mingli Zhu, Shaokui Wei, Li Shen, Yanbo Fan, and Baovuan Wu. Enhancing fine-tuning based back-
7.54	door defense with sharpness-aware minimization. In Proceedings of the IEEE/CVF International
/35	Conference on Computer Vision pp 4466–4477 2023a
736	
737	Mingli Zhu, Shaokui Wei, Hongyuan Zha, and Baoyuan Wu. Neural polarizer: A lightweight and
738	effective backdoor defense via purifying poisoned features. Advances in Neural Information
739	Processing Systems, 36, 2024.
740	
741	Xiaojin Zhu and Andrew B Goldberg. Introduction to semi-supervised learning. Springer Nature,
7/0	2022.
142	
743	Zixuan Zhu, Rui Wang, Cong Zou, and Lihua Jing. The victim and the beneficiary: Exploiting
744	a poisoned model to train a clean model on poisoned data. In Proceedings of the IEEE/CVF
745	International Conference on Computer Vision, pp. 155–164, 2023b.
746	
747	
748	
7/10	
743	
/50	
751	
752	
753	
754	

A EXTENDED DISCUSSIONS

758 A.1 EXTENDED RELATED WORKS

760 With the advance of clean subset extraction and backdoor detection, many works tries to split clean 761 subset from poison training sets. Zeng et al. (2023a) proposed detecting poisoned data by identifying 762 shifts in data distributions, which results in high prediction loss when training on the clean portion 763 of a poisoned dataset and testing on the corrupted portion. They solve a relaxed of the splitting 764 optimization problem with the help of a weight-assigning network. Although promising empirical results were presented, the proposed META-SIFT only guarantees a relatively small subset ((Zeng 765 et al., 2023a), page 10, Figure 5). As a result, META-SIFT still relies on effective downstream 766 defenses, such as NAD and ASD, included in our baselines, while also increasing the hyperparameter 767 search space. Pan et al. (2023) are motivated by the same distributional shift phenomenon and 768 proposed an effective splitting algorithm, ASSET. However, they assume that the defender has an 769 extra set of clean samples (named "base set" in (Pan et al., 2023)), which doesn't suit the background 770 of our paper, where no extra clean set is available. Plus, ASSET is faced with the same problem that 771 requires effective downstream defenses to conduct the defense. 772

In general, these works indeed provide valuable insights into the poisoned data splitting problem and could inspire our future research. However, they are faced with two major problems. 1) Cannot guarantee a 100% correct split that can be directly used for training; 2) Rely on an extra clean set which violates the constraints of our scenario.

777 778

785

786

787

A.2 ADDITIONAL BASELINES

We added two recent defense, VaB (Zhu et al., 2023b) and D-ST/D-BR (Chen et al., 2022b). The additional baselines are implemented based on the official implementation. We use CIFAR-10 as the dataset. Since the label-consistent attack is not consistently implemented, we use SIG as a clean label attack here. As shown in Table 7, VaB has the most competitive result against poison label attacks, but struggles to defend against SIG.

Table 7: The clean accuracy (CA%), attack success rate (ASR%), defense effective rating (DER%) and time cost (hours) of 2 additional backdoor defense methods and our BSD against 4 threatening backdoor attacks on CIFAR-10. The best and second best results are in **bold** and <u>underlined</u>.

Method	CA CA	BADNETS ASR	5 DER	CA E	BLENDEI ASR	DER	CA	WANET ASR	DER	CA	SIG ASR	DER	Avg DER	Time Cost
VAB D-ST	$\frac{94.0}{66.8}$	1.3 5.7	$\frac{98.9}{83.1}$	$\frac{94.2}{65.0}$	$\frac{1.1}{7.1}$	$\frac{98.6}{81.1}$	$\frac{93.6}{60.8}$	$\frac{1.7}{15.2}$	$\frac{99.1}{76.0}$	$\frac{94.0}{87.9}$	66.6 95.1	63.8 46.5	$\frac{90.1}{71.6}$	5.5 4.3
D-BR BSD(Ours)	87.5 95.1	0.8 <u>0.9</u>	95.9 99.6	83.0 94.9	80.7 0.8	53.2 98.8	16.9 94.5	14.6 0.8	54.3 99.6	85.7 93.8	$\frac{0.1}{0.0}$	<u>92.9</u> 97.0	74.1 98.7	3.2

794 795 796

797

798

799

800

801

802

803

804

A.3 DISCUSSION ON RESISTANCE TO CLEAN-LABEL ATTACK

The primary reason BSD can effectively resist clean-label backdoor attacks lies in how the MixMatch algorithm processes unlabeled data. Specifically, MixMatch applies a mixup operation that visually weakens the trigger and prevents it from being directly associated with the target label.

To better understand this phenomenon, we analyzed related semi-supervised defense methods like ASD. ASD report good performance against label-consistent attacks. In successful cases of ASD, poison samples were correctly classified into the poison pool, while failures often occurred when poison samples remained in the clean pool (Table 8).

Table 8: Number of poison samples in the clean pool of failed runs of ASD (CIFAR-10, LC attack, poisoning ratio 2.5%). The number of poison sample in clean pool exceeds 500 in average (poisoning ratio 2% in clean pool), which is the main reason that it fails to resist LC attack.

Еросн→	111	112	113	114	115	116	117	118	119	120	MEAN	STD
NUMOFPOISON(RUN 1)	405	567	683	321	695	280	303	595	391	910	515	208.4
NUMOFPOISON(RUN 2)	936	466	838	544	658	146	435	370	530	345	526.8	234.9

810 This led us to hypothesize that the key factor influencing clean-label attack defense is still the accurate 811 classification of poison samples into the poison pool. Further investigation revealed that the mixup 812 operation in MixMatch plays a critical role. MixMatch not only removes original labels but also 813 mixes multiple inputs on the image level, effectively weakening the visual impact of the trigger. We 814 visiualzed the mixed samples together with the argmaxed mixed label in Figure 4 to better present the mixup operation. 815



819 820 821 822

824

832

838

839

840

841

846 847 848

849

850 851

852

853



828 Figure 4: Visualization of the mixup operation in MixMatch. We take a random batch at the end of 829 training of our BSD against LC attack on CIFAR-10. Each group of three pictures has labeled data 830 on the left (original label), unlabeled data in the middle (predicted label by the model), and mixed data on the right (mixed label). 831

833 Because of $\alpha = 0.75$ in the default setting of MixMatch, λ' has a mathematical expectation of 834 approximately 0.78, the mixed inputs diminish the prominence of the trigger in x_u . We visualzed 835 the mixed samples together with to better present the mixup operation. Furthermore, we follow ASD to set a 5 times smaller λ_u value (15 vs. MixMatch's recommended 75), reducing the influence of 836 837 unlabeled data and further mitigating clean-label attacks.

Finally, to verify this hypothesis, we manually enforced a secure clean-poison split where no poison samples were included in the clean pool. Under this condition, MixMatch effectively nullified the impact of clean-label attacks, as shown in Table 9.

							1	υ				
METHOD	1	LC-CIFA	R	I	LC-GTSR	В	5	SIG-CIFA	R	NAR	cissus_C	IFAR
METHOD	CA	ASR	DER	CA	ASR	DER	CA	ASR	DER	CA	ASR	DER
NO DEFENSE	94.4	99.9	-	97.3	100.0	-	95.0	95.2	-	95.2	99.5	-
MIXMATCH*	94.2	0.0	99.9	97.4	0.0	100.0	94.6	0.0	97.4	94.5	0.0	99.3

Table 9: Train with MixMatch and secured clean pool against clean label attacks.

A.4 **ROBUSTNESS AGAINST DIFFERENT TARGETS**

We evaluated the robustness against different targets of our BSD in Table 10

Table 10: The clean accuracy (CA%), attack success rate (ASR%), and defense effective rating (DER%) of our BSD against 3 representitive backdoor attacks with different target labels on CIFAR-10.

854			BADNETS		1	BLENDED			WANET	
855	TARGET	CA	ASR	DER	CA	ASR	DER	CA	ASR	DER
856	0	95.0	0.8	99.6	95.0	0.4	99.0	91.9	0.3	99.0
857	1	94.9	0.5	99.8	94.9	0.5	98.9	94.2	0.3	99.8
001	2	95.1	0.8	99.6	94.7	0.9	98.7	90.9	0.7	98.3
858	3	95.1	0.9	99.6	94.9	0.8	98.8	94.5	0.8	99.6
850	4	95.0	0.2	99.9	94.8	0.6	98.9	92.4	0.2	99.3
000	5	95.1	1.7	99.2	95.0	0.5	98.9	91.9	0.4	98.9
860	6	95.1	0.6	99.7	93.9	0.6	98.8	92.6	0.3	99.3
861	7	94.7	0.3	99.8	92.6	0.4	98.2	90.3	0.0	98.3
	8	92.0	0.3	98.4	95.1	0.5	98.9	91.8	0.3	98.9
862	9	94.9	0.3	99.8	94.0	0.4	98.9	94.0	0.2	99.9
863	Avg	94.7	0.6	99.5	94.5	0.6	98.8	92.5	0.4	99.1

865 Clean Poiso Clean Poisor Clean Clean Clean 0.14 0.0 866 0.05 0.04 0.06 0,12 0.04 0,0 0.10 0.03 867 0.04 0,08 0.03 0.04 0,02 0.06 868 0.03 0.02 0,0 0.04 0.01 0.0 0.01 0.02 0.00 0.00 10 15 20 -70-10 870 871 0.0 Clean Clean Poison Clean 0.25 Clean Clean 872 0.0 0.06 0.06 0.20 0.04 0.0 873 0,05 0.04 0.15 0.03 0.04 874 0.03 0.03 0,10 0.02 875 0.02 0,0 0.01 0.0 0.0 876 0,00 - fo- - - 15 0,0 -- 50--75 100 12 877

Figure 5: Visiualization fo the effectiveness of OSS against BadNet-all2all attack.

A.5 ROBUSTNESS AGAINST ALL2ALL ATTACKS

All-to-all (all2all) attacks may pose challenges to certain components of our defense, particularly 884 OSS and selective drop. However, all2all attacks are not typically considered essential scenarios in 885 backdoor defense research currently (Li et al., 2021a; Huang et al., 2022; Zhu et al., 2023b; Guan 886 et al., 2024; Zhang et al., 2023), for the following reasons: 1) The increased number of trigger-target 887 pairs in all2all attacks requires significantly more training epochs for success. And all2all attacks reduce clean accuracy and exhibit slower convergence, making them easier to detect. (Huang et al. 889 (2024), Page 2: "As the number of classes increases, the accuracy and the attack success rate will 890 decrease.") 2) Research on all2all attacks remains limited (Li et al. (2022), Page 10: "However, there 891 were only a few studies on all-to-all attacks. How to better design the all-to-all attack and the analysis 892 of its properties remain blank."). 3) In practical applications, all 2all attacks do not allow attackers to 893 arbitrarily control predictions to specific targets, limiting their real-world threat.

Nevertheless, we still conducted supplementary experiments on BadNets with an all2all setting.

Attack setting: Following BadNets, with $y_t = (y+1) \Re n_c$, where n_c is the number of classes.

B97 Defense setting: To handle multiple target labels, BSD incurs additional computational costs by iterating through all classes as pseudo-targets during OSS. Clean indices from each pseudo-target are intersected to form the final OSS result. Additionally, we early stop at Stage 2 to avoid meaningless cost in Stage 3.

Since all-to-all attacks do not fundamentally change the nature of poison-label attacks, OSS remains effective for each individual classes. We visualized OSS spliting results in Figure 5, which reveals effective separation of clean samples of OSS. The CA, ASR, and DER performance are presented in Table 11, demonstrating a significant DER improvement compared to baseline methods. Notably, while BSD's ASR increases under all-to-all attacks, it effectively limits the attack success rate to the level of random prediction $(1/n_c = 10\%)$.

In conclusion, our BSD method remains effective against all-to-all attacks. Furthermore, the OSS
 module can serve as a highly effective component for identifying clean samples in other backdoor defense methods.

910 911 912

012

864

878

879 880

Table 11: The clean accuracy (CA%), attack success rate (ASR%), and defense effective rating
(DER%) of ASD and our BSD against BadNets-all2all on CIFAR-10.

O DEFENSE	91.8	93.8	-
ASD SD (OURS)	70.2	2.4	84.9 91.3
	O DEFENSE ASD SD (OURS)	O DEFENSE 91.8 ASD 70.2 SD (OURS) 91.2	O DEFENSE 91.8 93.8 ASD 70.2 2.4 SD (OURS) 91.2 10.5

918 A.6 PSEUDO TARGET APPROXIMATION TEST

Pseudo target approximation test on GTSRB. We evaluated the approximation of y_t on the GTSRB dataset with various ground truth target labels, as shown in Table 12 (using the alternative approximation method described in Appendix E.3). The results demonstrate that y_t was successfully approximated for all of the first 10 classes in GTSRB.

Table 12: Testing the y_t approximation on different target labels (the first 10 classes) on GTSRB.

	\mathcal{O}	00 1	1			\mathcal{O}	`		/	
ATTACK	0	1	2	3	4	5	6	7	8	9
BADNETS	\checkmark	\checkmark	\checkmark	\checkmark	~	\checkmark	\checkmark	\checkmark	\checkmark	~
Blended	\checkmark	\checkmark	\checkmark	\checkmark	\checkmark	\checkmark	\checkmark	\checkmark	\checkmark	\checkmark
LC	\checkmark	\checkmark	\checkmark	\checkmark	\checkmark	\checkmark	\checkmark	\checkmark	\checkmark	\checkmark

Performance under forced incorrect pseudo target label. We conducted interesting additional tests by forcing y_t to be assigned to an incorrect class and observed the model's performance (on CIFAR-10, against BadNets). As illustrated in Figure 6, BSD retained partial defensive capabilities even when the pseudo-label was deliberately set incorrectly. In most cases presented, BSD managed to purify the model successfully, leveraging the loss-guided splitting based on the Altruistic model.

It is worth noting that in experiments where the ground truth target class was 5 (dog), forcibly setting the pseudo-label to 3 (cat) led to a significant failure of the defense. This may be attributed to the inherent difficulty in distinguishing between these two classes. Furthermore, when faced with broader attack scenarios and dataset settings, relying solely on loss statistics may not be sufficient to ensure effective defense. Fortunately, our experiments demonstrate the strong robustness of the proposed Pseudo Target Approximation method. The OSS mechanism functioned as expected, enabling a resilient bi-perspective defense under challenging conditions.



Figure 6: The clean accuracy (CA%), attack success rate (ASR%), and robust accuracy (RA%) of BSD when Forcing the pseudo target from x(ground truth) to 3.

B Algorithm outline

973 974 975

976 977 978

972

The pseudocode of the proposed method BSD is listed as Algorithm 1.

Algorithm 1 Pseudocode for BSD

```
979
            Input: Poisoned training set D; main model f; main model warm-up ends at epoch T_1, main model
980
            training stage2 ends at epoch T_2, main model training stage3 ends at epoch T_3, max clean pool ratio
981
            \alpha, OSS split ratio \beta.
982
            Output: Clean model f_{\theta'}
983
             1: # Initialization & warm-up
984
             2: Initialize the weights of f as \theta
985
             3: Generate an altruistic model g having the same architecture as f, initialize the weights as \varphi
986
             4: # Prepare for ALS
             5: for i = 1 to 25 do
987
                     for each sample (x, y) in \mathcal{D} do
988
             6:
             7:
                         loss \leftarrow \mathcal{L}_{ce}(x, y, g_{\varphi})
989
                         loss \leftarrow sign(loss - 0.5) \times loss # Default LGA
             8:
990
             9:
                         \varphi \leftarrow \varphi - \nabla_{\varphi} loss
991
            10:
                     end for
992
            11: end for
993
            12: # Prepare for OSS
994
            13: Set y_t as the most frequent class among the 1% lowest \mathcal{L}_{ce}(g_{\varphi}) samples in \mathcal{D}
995
            14: Calculate \mathcal{D}_{t} and \mathcal{D}_{nt} with y_{t} according to Section 3.2
996
            15: # Main Training Loop
997
            16: while T < T_3 do
998
            17:
                     if T < T_1 then
                         # Data used for the main model warm-up
            18:
999
            19:
                         \mathcal{D}_c \leftarrow \mathcal{D}_{\mathrm{nt}}
1000
            20:
                     else if T = T_1 then
1001
            21:
                         # Pool initialization using ALS and OSS
1002
                         \mathcal{D}_c \leftarrow \mathcal{D}_{als} \cup \mathcal{D}_{oss}
            22:
1003
            23:
                     else if T_1 + 10 \leq T < T_2 then
1004
            24:
                         # Pool update based on loss discrepancy of f_{\theta} and g_{\varphi}, enabling class completion
1005
                         T' \leftarrow \frac{T - T_1 - 10}{T_2 - T_1 - 10} T_2
            25:
                         Current clean ratio \alpha_T \leftarrow \beta + (\alpha - \beta) \times (1 - \cos(\pi \times T'/T_2))/2
            26:
            27:
                         Set \alpha as \alpha_T in equation 8
1008
            28:
                         Calculate \mathcal{D}_{\tilde{c_1}} according to equation 8
1009
                      \begin{aligned} \mathcal{D}_c \leftarrow \mathcal{D}_{\tilde{c_1}} \\ \text{else if } T \geq T_2 \text{ then} \end{aligned} 
            29:
1010
            30:
1011
                         # Pool update based on loss discrepancy of f_{\theta} and g_{\varphi}, enabling class completion and
            31:
1012
                         selective drop
1013
            32:
                         Current clean ratio \alpha_T \leftarrow \alpha
            33:
                         Set \alpha as \alpha_T in equation 10
1014
            34:
                         Calculate \mathcal{D}_{\tilde{c}_2} according to equation 10
1015
                         \mathcal{D}_c \leftarrow \mathcal{D}_{\tilde{c_2}}
            35:
1016
                     end if
            36:
1017
            37:
                     \mathcal{D}_p \leftarrow \mathcal{D} \setminus \mathcal{D}_c
1018
                     # Models updating
            38:
1019
            39:
                     \theta \leftarrow \theta - \nabla_{\theta} \mathcal{L}_{\text{semi}} # Train the model on \mathcal{D}_c (labeled) and \mathcal{D}_p by semi-supervised learning
1020
            40:
                     if T < T_2 then
1021
            41:
                         \varphi \leftarrow \varphi - \nabla_{\varphi} \mathcal{L}_{ce} # Train the altruistic model by supervised learning
            42:
                     end if
1023
            43: end while
1024
```

1026 C IMPLEMENTATION DETAILS

1028 C.1 ENVIRONMENTS

We run all the experiments using PyTorch on a Linux server with an AMD EPYC 7H12 64-core
Processor, 256GB RAM, and 8× NVIDIA GeForce RTX 3090 GPU.

1033 C.2 Illustration of the poisoned samples

Figure 7 illustrates the seven attack types used in this study, displaying both the original and poisoned images along with the corresponding trigger patterns. For attacks involving a different trigger in the Imagenet dataset, the specific trigger is also shown at the bottom.



Figure 7: Illustation of the backdoor attacks. We present the examples on CIFAR-10, alternative triggers (if used) on Imagenet are shown at the bottom.

1058 1059

1061

1032

0 C.3 ATTACK SETTINGS

Training settings. For all the attack implementations, we follow that in ASD (Gao et al., 2023). On 1062 the CIFAR-10 and GTSRB datasets, we perform backdoor attacks on ResNet-18 for 200 epochs with 1063 batch size 128. We adopt the stochastic gradient descent (SGD) optimizer with a learning rate of 1064 0.1, momentum of 0.9, and weight decay 5×10^{-4} . The learning rate is divided by 10 at epoch 100 and 150. For attacks not achieving reported performance in ASD (Gao et al., 2023), we continue the training for another 100 epochs, and the learning rate is divided by 10 at epoch 200 and 250. On 1067 the Imagenet (Deng et al., 2009) dataset, we train ResNet-18 for 90 epochs with batch size 256. We 1068 utilize the SGD optimizer with a learning rate of 0.1, momentum of 0.9, and weight decay 1×10^{-4} . 1069 The learning rate is decreased by a factor of 10 at epoch 30 and 60. The image resolution will be 1070 resized to $224 \times 224 \times 3$ before attaching the trigger pattern.

Settings for BadNets. As suggested by Gu et al. (2017); Huang et al. (2022); Gao et al. (2023), we set a 2×2 square on the upper left corner as the trigger pattern on CIFAR-10 and GTSRB. For ImageNet and VGGFace2, we use a 32×32 apple logo on the upper left corner. The poisoning rate ρ is set to 0.05(5%).

Settings for Blended. Following Chen et al. (2017); Huang et al. (2022); Gao et al. (2023), we choose "Hello Kitty" pattern on CIFAR-10 and GTSRB and the random noise pattern on ImageNet and VGGFace2. The blend ratio is set to 0.1. The poisoning rate ρ is set to 0.05(5%).

Settings for WaNet. Following Gao et al. (2023); Huang et al. (2022), we directly use the default warping-based operation to generate the trigger pattern. For CIFAR-10 and GTSRB, we set the noise

rate ρ_n to 0.2, control gird size k as 4, and warping strength s as 0.5. For Imagenet, we use the same noise rate, but a larger grid size k = 224, and a warping strength s = 1.

Settings for Label-Consistent Attack. Following Gao et al. (2023); Huang et al. (2022); Turner et al. (2019), the noisy versions of samples are generated using adversarially trained models. The PGD parameters are as follows: for PGD training: $\epsilon = 16$, $\alpha = 2$, steps = 7, and the pixel range is [0, 255]; for PGD attack: $\epsilon = 16$, $\alpha = 1.5$, steps = 30, with the same pixel range [0, 255]. The same trigger used in BadNets is applied for LC attacks, and the poison ratio is set at 25

Settings for Refool. Following Li et al. (2021a); Liu et al. (2020), we randomly choose 5,000 images from PascalVOC (Everingham et al., 2015) as the candidate reflection set \mathcal{R}_{cand} and randomly choose one of the three reflection methods to generate the trigger pattern during the backdoor attack.

Settings for SIG. Following Li et al. (2021a); Barni et al. (2019), we adopt the same sinusoidal pattern in ABL as the trigger and set the poisoning rate to match LC, as SIG is a clean-label attack.

Settings for Narcissus. We also incorporate the recent attack proposed by Zeng et al. (2023b), which is another clean-label attack. The parameter settings for generating the Narcissus trigger pattern are as follows: the ℓ_{∞} ball bound is set to 16/255, the surrogate model is trained for 200 epochs with an initial learning rate of 0.1 and a warm-up period of 5 rounds. The trigger-generation learning rate is 0.01, and the generation process lasts for 1000 rounds. The poisoning rate is the same as LC, given that Narcissus is also a clean-label attack.

- 1099
- 1100 C.4 DEFENSE SETTINGS

1102 Settings for FP. Following Gao et al. (2023), we set two steps of FP Liu et al. (2018) (i.e., pruning 1103 and fine-tuning) as follows. (1) We randomly select 5% clean training samples as the local clean samples and forward them to obtain the activation values of neurons in the last convolutional layer. 1104 The dormant neurons on clean samples with the lowest α % activation values will be pruned. (2) The 1105 pruned model will be fine-tuned on the local clean samples for 10 epochs. In particular, the learning 1106 rate is set as 0.01, 0.01, 0.1 on CIFAR-10, GTSRB, and ImageNet. Unless otherwise specified, other 1107 settings are the same as those used by Liu et al. (2018). For the hyper-parameters of FP, we search 1108 for the best results by adjusting the pruned ratio $\alpha\% \in 20\%, 30\%, 40\%, 50\%, 60\%, 70\%, 80\%, 90\%$. 1109 In addition, we add another default setting in backdoorbench (Wu et al., 2022). 1110

Settings for NAD. NAD (Li et al., 2021b) is also a post trianing method that repairs the backdoored 1111 model and needs 5% local clean training samples. We set the two steps of NAD as follows: (1) Use 1112 the local clean samples to fine-tune the backdoored model for 10 epochs. Specially, the learning rate 1113 is set as 0.01, 0.01, 0.1 on CIFAR-10, GTSRB, and ImageNet. (2) The fine-tuned model and the 1114 backdoored model will be regarded as the teacher model and student model to perform the distillation 1115 process. Unless otherwise specified, other settings are the same as those used by Li et al. (2021b). 1116 For the sensitive hyper-parameter β , we find the search space used by Gao et al. (2023) too small. 1117 We search for the best results by adjusting the hyper-parameter β from 500, 1000, 1500, 2000, 2500, 1118 5000, 7500, 1e5, 1e6, 1e7, 1e8, 1e9, 1e10, 1e11. In addition, we add another default setting in backdoorbench (Wu et al., 2022). 1119

1120 Settings for ABL. ABL (Li et al., 2021a) contains three stages: (1) To obtain the poisoned samples, 1121 ABL first trains the model on the poisoned dataset for 20 epochs by LGA loss and isolate 1% training 1122 samples with the lowest loss. (2) Continue to train the model with the poisoned dataset after the 1123 backdoor isolation for 70 epochs. (3) Finally, the model will be unlearned by the isolation samples 1124 for 5 epochs. The learning rate is 5e-4 at the unlearning stage. Unless otherwise specified, other 1125 settings are the same as those used by Li et al. (2021a). ABL is sensitive to the hyper-parameter γ in LGA loss. We search for the best results by adjusting the hyper-parameter γ from 0, 0.1, 0.2, 0.3, 0.4, 1126 0.5, In addition, we add another default setting in backdoorbench (Wu et al., 2022). 1127

Settings for DBD. DBD (Huang et al., 2022) contains three independent stages: (1) DBD uses
SimCLR to perform the self-supervised learning for 1,000 epochs. (2) Freeze the backbone and
fine-tune the linear layer by supervised learning for 10 epochs. (3) Adopt the MixMatch to conduct
the semi-supervised learning for 200 epochs on CIFAR-10 and GTSRB for 90 epochs on ImageNet
and VGGFace2. Unless otherwise specified, other settings are the same as those used by Huang et al.
(2022). Since DBD is a relatively stable backdoor defense and not sensitive to its hyper-parameter,
we only add another group of default setting in backdoorbench (Wu et al., 2022).

1134 Settings for ASD. We follow the exact settings for ASD as suggested by Gao et al. (2023). To name 1135 a few settings, we adopt MixMatch as the semi-supervised learning framework and use the Adam 1136 optimizer with a learning rate of 0.002 and a batch size of 64 for the semi-supervised training. The 1137 temperature T is set to 0.5, and the weight of the unsupervised loss λ_{μ} is set to 15. The training 1138 stages are defined as follows: $T_1 = 60$, $T_2 = 90$, and $T_3 = 120$ for CIFAR-10 and ImageNet, while $T_3 = 100$ for GTSRB. Similarly, other parameters are the same as used by Gao et al. (2023) as well. 1139 1140 For our BSD, we adopt the MixMatch Berthelot et al. (2019b) semi-supervised training framework for 1141 the main model, following Decoupling-based Defense (DBD) and Adaptive Splitting-based Defense 1142 (ASD). The semi-supervised learning parameters align with ASD, including 1024 training iterations, 1143 a temperature of 0.5, a ramp-up length of 120, and a learning rate of 0.002. The altruistic model 1144 undergoes a warm-up phase with 25 epochs, utilizing the Adam optimizer, Cross Entropy loss, with a learning rate of 0.001. The default warm-up epochs for the main model in OSS are set to 20 (followed 1145 by a 10-epoch training on the initialized pools)($T_1 = 20$), with a default β of 0.2. Class completion 1146 training spans 60 epochs ($T_2 = 90$), and selective dropping training spans 30 epochs ($T_3 = 120$). 1147 The altruistic model update uses the same loss and optimizer as in the warm-up on CIFAR-10 and 1148 Imagenet for efficiency, on lightweight datasets like GTSRB, we use the same semi-supervised loss 1149 and optimizer as the main model for better performance. The clean pool ratio α follows a sinusoidal 1150 growth curve during class completion training, starts at β , and reaches an upper limit of $\alpha = 0.6$ at 1151 the end of the class completion stage, after which it remains fixed: 1152 $T' = \frac{T - T_1 - 10}{T_2 - T_1 - 10} T_2$ 1153 1154 (11) $\alpha_T = \beta + (\alpha - \beta) \times (1 - \cos(\pi \times T'/T_2))/2$ 1155 1156 1157 The baselines are implemented using: 1158 1159 • BackdoorBench (Wu et al., 2022); 1160 • BackdoorBox (Li et al., 2023b); 1161 • Github repositories of corresponding papers. 1162 1163 We greatly appreciate these outstanding works. 1164 1165 C.5 DEFINITION OF DER 1166 1167 Defense Effectiveness Rating (DER) (Zhu et al., 2023a) is a comprehensive measure that considers 1168 both ACC and ASR: 1169 1170 $DER = [max(0, \Delta ASR) - max(0, \Delta ACC) + 1]/2,$ (12)1171 where ΔASR denotes the decrease of ASR after applying defense, and ΔACC denotes the drop 1172 in ACC following the defense. Higher ACC, lower ASR and higher DER indicate better defense 1173 performance. 1174 1175 1176 C.6 DETAILS ABOUT SEMI-SUPERVISED LOSS 1177 Semi-supervised learning (Berthelot et al., 2019a;b; Sohn et al., 2020; Xie et al., 2020; Zhu & 1178 Goldberg, 2022) studies how to leverage a training dataset with both labeled data and unlabeled data 1179 to obtain a model with high accuracy. In addition to its application in normal training, semi-supervised 1180 learning also serves as a powerful means for the security of DNNs (Alayrac et al., 2019; Carmon 1181 et al., 2019; Huang et al., 2022). 1182 Here we adopt the MixMatch (Berthelot et al., 2019b). Given a batch $\mathcal{X} \subset \mathcal{D}_C$ of labeled samples, 1183 and a batch $\mathcal{U} \subset \mathcal{D}_P$ of unlabeled samples, MixMatch generates a guessed label distribution \tilde{q} for 1184 each unlabeled sample $u \in \mathcal{U}$ and adopts MixUp to augment \mathcal{X} and \mathcal{U} to \mathbf{X}' and \mathbf{U}' . The supervised 1185

- 1186 loss \mathcal{L}_s is defined as:
- 1187 $\mathcal{L}_{s} = \sum_{(x,q)\in\mathcal{X}'} \mathrm{H}\left(p_{x},q\right),\tag{13}$

where p_x is the prediction of x, q is the one-hot label and $H(\cdot, \cdot)$ is the cross-entropy loss. The unsupervised loss \mathcal{L}_u is defined as:

 $\mathcal{L}_{u} = \sum_{(u,\bar{q})\in\mathcal{U}'} \|p_{u} - \bar{q}\|_{2}^{2}, \qquad (14)$

where p_u is the prediction of u.

1195 Finally, the MixMatch loss can be defined as:

$$\mathcal{L} = \mathcal{L}_s + \lambda \cdot \mathcal{L}_u, \tag{15}$$

where λ is a hyper-parameter for trade-off, we adopt the same $\lambda = 15$ as in ASD.

1199

1201

1196 1197

D SUPPLEMENTARY INFORMATION OF THE BACKGROUND

1202 D.1 SUPPLEMENTARY OVERVIEW OF BACKDOOR ATTACK RESEARCH

The common implementation of backdoor attacks is realized by injecting a few poisoned samples into the training dataset, i.e., data-poisoning-based backdoor attacks, inducing the model to build a link between the trigger (i.e., a visual particular pattern) and target class (Gu et al., 2017). Thus the model consistently outputs the target label once the trigger is attached to the inputs in the inference stage.

Poison-label backdoor attacks are currently the most common attack paradigm, where the trigger pattern in the poisoned samples is directly connected to the target class by relabeling, inducing the model to treat the trigger as a decision-making feature of the target class. Recent research (Hu et al., 2022; Li et al., 2020; Qi et al., 2021) focuses on more invisible trigger designs through generative models and feature space optimizations, as well as exploring backdoor attacks in wider tasks like natural language processing.

1214

1223

1224 1225

1226

1227 1228

1229 1230

1215 D.2 ILLUSTRATION OF THE MODEL COLLAPSE

As presented in Figure 8, the splitting-based defenses (loss-guided ones specifically) encounter two kinds of model collapse. In backdoor overfitting collapse, poison samples take effect and have low loss values, which consistently corrupt the clean pool and lead to a backdoored model. Likewise, in class underfitting collapse, the rareness of certain classes will lead to higher loss values, making them less chosen to be clean samples, which forms a vicious cycle. Note that these two collapses are common in other categories of defenses as well, while it's more explainable in splitting-based defense.



Figure 8: Typical model collapses in data-splitting backdoor defenses. I: Misclassification of low-loss poisoned samples as clean leads to a steady increase in the poisoned sample proportion until 100% ASR. II: Higher losses for challenging classes reduce their presence in the clean pool, rendering the model unable to predict samples from those categories.

1235 1236

1240

1237 E MORE EXPERIMENTAL RESULTS

1239 E.1 Illustration of pool update

To showcase the healthy clean pool acquired by our BSD, we plot the number of poison samples in the clean pool at each training epoch, as well as reveal the accumulated number of poison samples.

As shown in Figure 9, our BSD generally have fewer poison samples in the clean pool during training, with both the number of poisoned samples and the cumulative number of samples smaller than that fo
ASD under different poisoning rates.

In addition, we plot the loss/distance distribution of samples of our BSD in Figure 10. In Figure 10.(a) and Figure 10.(b), the main mechanisms, i.e., ALS and OSS for the pool initialization, successfully distinguished the poison samples. In Figure 10.(c), the poison samples are highlighted by the high loss discrepancy between the main model and the altruistic model. The final result shown in Figure 10.(d) reveals the high CAs (clean sample all having low loss values) and low ASRs (poison samples all having high loss values) of BSD.



Figure 9: The number of poison samples in the clean pool of BSD and ASD at each epoch, the accumulated number in the dotted line. The subplots are the results on CIFAR-10, $\rho = 0.01, 0.05, 0.10, 0.20$.



Figure 10: The split visualization of BSD on GTSRB against the BadNet attack. (a) the loss distribution on the altruistic model after the ALS warm-up; (b) the distance distribution on the main model after the OSS warm-up; (c) the loss discrepancy at the last epoch; (d) the loss distribution on the main model at the last epoch.

1282 1283 E.2 INFLUENCE OF DIFFERENT SETTINGS IN OSS

1262

1263 1264 1265

1280 1281

1284 Ablation on distance metric of OSS. We investigate the influence of the number of runs for the 1285 warm-up and three different distance calculations of OSS as shown in Figure 11. For the distance 1286 calculation, we take three approaches, i.e., the minimal, the maximal, and the mean ℓ_2 distance 1287 to feature clusters of each non-target category. Intuitively, we set the minimal distance by default, because the poisoned samples we consider are characterized by being far away from all existing 1288 cluster centers, thus maximal and mean distances may misjudgment two categories whose original 1289 clustering centers are far apart from each other as poisoned samples. Whereas, all three approaches 1290 exhibit good separation under the default 20-epoch warm-up. 1291

1292 Ablation on warm-up epochs of OSS (T_1) . Concerning the number of warm-up epochs, we 1293 investigate the score distribution of OSS under the min-distance calculation. As illustrated in 1294 Figure 11, the result exhibits poor separation with an insufficient warm-up. As the number of epochs 1295 goes up, it has a certain effect when the number of epochs equals 10, and perfectly separates some 1296 benign samples with larger epochs. Figure 11 is the complete result of the ablation study on different settings of the warm-up process of OSS. It verifies the effectiveness of all three distance metrics. Intuitively, the model should have a more separable and reliable initialization of the two pools with a long warm-up, whereas the result of a 40-epoch warm-up (especially when using the mean distance) violates this intuition by exhibiting less satisfying separation. A potential reason is that the model overfitted the non-target classes, thus the poison samples have less similarity to them.



Figure 11: The ablation results on the number of warm-up epochs and different distance calculation methods for OSS.

In general, the default setting of BSD is a suitable choice.

1332 1333 1334

1335

1330 1331

E.3 INFLUENCE OF DIFFERENT SETTINGS IN ALS

1336 Ablation study on warm-up epochs of the altruistic model. The warm-up of the altruistic model 1337 will determine the correctness of y_t . Instead of directly assuming y_t being known to the defender, 1338 we explored lightweight methods to approximate it. Zhu et al. (2024) uses the most frequent second 1339 likely prediction as approximated y_t , Gao et al. (2024) uses a likewise energy score, which is also 1340 effective. Li et al. (2021a) proposes to isolate the poison samples through a local gradient ascent 1341 process. If the detection precision exceeds 50%, it indicates that more than half of the isolated samples are poison samples, thus we can obtain y_t . Although the experimental results presented by Li 1342 et al. (2021a) in their Figure 7, page 16 has already verified a more than 50% against most common 1343 attacks, we further check its robustness to the warm-up epochs in Table 13. 1344

1345 Alternative method for approximating y_t For unseen failures that the local gradient ascent (Li 1346 et al., 2021a) can not correctly approximate y_t , we provide an alternative method for approximating 1347 y_t against new backdoor attacks that may appear in the future. Here we follow Zhu et al. (2024) to 1348 use the most frequent second likely prediction, i.e., $y_t = \operatorname{argsort}(-\log it)[1]$, where logit means the 1349 logit output of a DNN. However, this prediction could be unstable, we further statistics the predicted y_t in each warm-up epoch and use the majority as the final prediction of y_t . The effectiveness of the

WARM UP FROQUE	BAD	Net	BLEN	DED
WARM-UP EPOCHS	NUM POISON	y_t	NUM POISON	$y_{-}t$
5	0	WRONG	0	WRONG
15	461	CORRECT	457	CORRECT
25	370	CORRECT	344	CORRECT
35	85	CORRECT	126	WRONG
45	177	CORRECT	114	CORRECT

Table 13: The prediction of y_t under different warm-up epochs. The 'num' represents the number of poison samples in the isolated set.

alternative method is shown in Figure 12, where all the final majority predictions of y_t are the same $(y_t = 3)$, which is the ground truth target label.



Figure 12: The result of the alternative method for y_t approximation. In the default 25 epochs of warm-up, we count the pred y_t at each epoch and the most pred y_t by that epoch respectively. The experiment is conducted on CIFAR-10, against BadNet, Blended, WaNat, and LC.

E.4 PERFORMANCE UNDER NO ATTACKS

Most backdoor defense research focuses on performance under attack, while it is concerning that these defenses may degrade model performance in the absence of attacks. Therefore, we evaluated the performance of BSD in scenarios without attacks. There are no poisoned samples in the training set, we test the clean and poisoned samples (BadNets trigger) for inference. As shown in Table 14, it is worth noting that even in the absence of attacks, there can be a low attack success rate (ASR), where these samples are just being misclassified to the target label. As Table 14 reveals, our method achieves a lower ASR compared to the baseline, effectively suppressing natural backdoors. Additionally, our method shows significant improvements in accuracy over the baseline.

attacks.				
WETHOD	CIFA	.R-10	GT	SRB
METHOD	ACC	ASR	ACC	ASR

Table 14: The clean accuracy (CA%) and attack success rate (ASR%) of BSD and ASD under no

METHOD	CIFA	R-10	GTSRB				
METHOD	ACC	ASR	ACC	ASR			
NO DEFENSE	95.3	1.9	97.7	0.2			
ASD	93.2	1.8	96.6	0.1			
BSD (OURS)	94.9	0.6	97.6	0.0			

RESISTANCE TO POTENTIAL ADAPTIVE ATTACKS E.5

In the above experiments, we assume that attackers have no information about our backdoor defense. In this section, we consider a more challenging setting, where the attackers know the existence of our defense and can construct the poisoned dataset with an adaptive attack.

Threat model for the attackers. Following existing work (Gao et al., 2023; Chen et al., 2017; Gu et al., 2017; Turner et al., 2018), we assume that the attackers can access the entire dataset and know

the architecture of the victim model. However, the attackers can not control the training process after
 poisoned samples are injected into the training dataset.

Method for adaptive attack. Our method initializes the clean pool using bi-perspective splitting through OSS and ALS, which separate poisoned samples based on semantic and loss statistics, respectively. In general, there is a contradiction between increasing the loss values of poisoned samples (to bypass ALS) and achieving backdoor objectives. Furthermore, maintaining high semantic similarity to the target class (to bypass OSS) adds to the complexity. To craft such a trigger pattern that satisfies the above objectives, we use a PGD optimization to search for an average noise (among non-target classes) that is semantically close (judged by a proxy model) to the target class (to bypass OSS). Meanwhile, we control the ℓ_{∞} ball bound as 8/255 and the poisoning rate as 0.01 to prevent it from being an obvious trigger that will be easily fitted (to bypass ALS).

Settings. We conduct experiments on CIFAR-10 with the following parameters: number of iterations, 15; step size, 1.5/255; perturbation magnitude, 8/255; trigger size, 32×32; and poisoning rate, 0.01.
Although the attacker is assumed to have no knowledge of the model structure, we adopt a more challenging setting where the adversary uses the same model structure as the proxy model.

Results. The Clean Accuracy (CA) and Attack Success Rate (ASR) of this adaptive attack are
94.422% and 2.421%, respectively. While the ASR is slightly higher than that of other attacks on
CIFAR-10, our defense clearly demonstrates strong resistance to the adaptive attack. Furthermore,
when we increase the poisoning rate to 0.2 (20%), the CA and ASR remain at 91.040% and 0.903%,
respectively, which is still within an acceptable range.

1425 E.6 SEARCHING THE BEST RESULT OF BASELINES

Notably, some baseline methods are sensitive to their hyper-parameter settings. The results reported
in Table 1 represent their best performance obtained through grid search, as outlined in ASD (Gao
et al., 2023). Similarly, for the additional attack settings, the results in Table 2 and Table 3 are based
on their best outcomes (ranking on DERs) after grid search. For DBD, which is not sensitive to
parameters, we report the best result using the default settings from BackdoorBench and the same
configuration as in ASD.

Table 15: Grid search for FP against additional attacks on ResNet18 (Default represents the result under the default setting provided by backdoorbench).

Dumo	L	C	51	ſĠ	Ref	OOL	NARC	CISSUS
KATIO	CA	ASR	CA	ASR	CA	ASR	CA	AS
DEFAULT	87.1	24.4	87.1	60.8	86.5	23.0	87.2	63
0.1	87.3	79.8	87.2	81.4	86.8	25.6	87.4	72
0.2	87.0	59.4	87.0	82.4	86.5	28.0	86.7	77
0.3	86.7	51.6	87.0	83.2	86.5	27.6	87.2	73
0.4	87.0	49.0	87.2	85.7	86.4	28.8	87.2	77
0.5	85.7	67.3	86.2	86.1	85.2	29.6	86.7	79
0.6	86.0	73.6	86.7	90.1	85.6	31.6	86.6	80
0.7	86.3	80.0	86.8	88.5	86.1	31.0	87.1	79
0.8	87.0	74.9	86.9	87.7	86.4	28.3	87.3	78
0.9	87.3	62.0	87.0	80.3	86.5	25.2	87.6	72

Table 16: Grid search for FP against representative attacks on Mobilenetv2 (Default represents the result under the default setting provided by backdoorbench).

D ATTO	BAD	NET	BLE	NDED	WA	Net
KAIIO	CA	ASR	CA	ASR	CA	ASR
DEFAULT	77.9	8.3	75.9	30.8	82.2	2.4
0.1	80.8	58.3	78.4	57.9	79.3	3.0
0.2	79.6	84.0	78.6	56.1	80.1	3.8
0.3	80.6	99.8	79.0	51.8	77.2	5.8
0.4	79.5	73.5	77.8	53.7	77.3	4.4
0.5	79.5	10.9	77.1	70.0	78.3	1.3
0.6	79.7	19.3	76.6	67.3	78.6	2.0
0.7	79.1	61.7	76.3	65.7	79.1	1.9
0.8	79.7	12.7	77.3	66.1	78.3	1.2
0.9	80.4	99.2	78.4	62.6	79.3	2.5

Table 17: Grid search for NAD against additional attacks on ResNet18 (Default represents the result under the default setting provided by backdoorbench).

DET	LC		S S	IG	Ref	OOL	NARC	CISSUS
BETA	CA	ASR	CA	ASR	CA	ASR	CA	ASF
DEFAULT	86.2	69.7	86.0	83.6	85.3	49.3	86.1	84.9
100	87.3	98.7	87.1	95.9	86.5	66.9	86.8	93.
500	86.2	69.7	86.0	83.6	85.3	49.3	86.1	84.9
1000	86.2	69.7	86.0	83.6	85.3	49.3	86.1	84.9
1500	86.2	69.7	86.0	83.6	85.3	49.3	86.1	84.
2000	86.2	69.7	86.0	83.6	85.3	49.3	86.1	84.
2500	86.2	69.7	86.0	83.6	85.3	49.3	86.1	84.
5000	86.2	69.7	86.0	83.6	85.3	49.3	86.1	84.
7500	86.2	69.7	86.0	83.6	85.3	49.3	86.1	84.
1.E+04	86.2	69.7	86.0	83.6	85.3	49.3	86.1	84.
1.E+05	86.2	69.7	86.0	83.6	85.3	49.3	86.1	84.
1.E+06	86.2	69.9	85.8	83.3	85.3	49.3	86.2	84.
1.E+07	85.9	50.5	85.7	84.5	85.5	46.5	86.5	81.
1.E+08	86.0	73.9	85.8	88.7	85.6	42.9	86.1	83.
1.E+09	85.3	68.2	85.7	87.0	85.6	44.8	86.5	84.
1.E+10	85.9	69.5	85.8	83.0	85.4	47.7	86.0	86.
1.E+11	85.9	65.6	85.7	86.3	85.6	42.5	86.4	83.

Table 18: Grid search for NAD against representative attacks on Mobilenetv2 (Default represents the result under the default setting provided by backdoorbench).

Dem	BADNET		BLEN	NDED	WANET	
BETA	CA	ASR	CA	ASR	CA	ASR
DEFAULT	78.5	11.7	76.2	51.6	81.1	4.2
100	79.7	99.1	79.5	56.8	77.3	3.4
500	78.5	11.7	76.2	51.6	81.1	4.2
1000	78.5	11.7	76.2	51.6	81.1	4.2
1500	78.5	11.7	76.2	51.6	81.1	4.2
2000	78.5	11.7	76.2	51.6	81.1	4.1
2500	78.5	11.7	76.2	51.6	81.1	4.1
5000	78.5	11.7	76.2	51.6	81.1	4.1
7500	78.5	11.7	76.2	51.6	81.1	4.1
1.E+04	78.5	11.7	76.2	51.6	81.1	4.1
1.E+05	78.6	29.1	76.1	51.3	79.8	4.0
1.E+06	78.5	23.0	75.9	49.4	81.5	3.2
1.E+07	79.3	25.4	76.8	47.6	81.0	3.8
1.E+08	78.8	14.8	76.8	58.9	81.0	3.2
1.E+09	78.9	17.9	76.0	46.0	80.7	5.4
1.E+10	79.2	25.0	76.5	51.8	81.6	4.1
1.E+11	78.8	41.5	76.7	56.9	81.3	4.3

Table 19: Grid search for ABL against additional attacks on ResNet18 (Default represents the result under the default setting provided by backdoorbench).

CURREL	LC		SIG		Refool		NARCISSUS	
GAMMA	CA	ASR	CA	ASR	CA	ASR	CA	ASI
DEFAULT	71.1	4.3	37.5	0.0	63.3	94.0	62.7	0.4
0.0	81.2	7.3	81.2	74.6	79.4	86.4	80.7	76.
0.1	81.2	6.0	79.7	41.5	72.0	93.5	80.3	35.
0.2	80.3	3.8	73.9	16.5	76.3	82.0	79.4	32.
0.3	78.1	0.9	76.1	14.4	76.0	86.5	79.3	7.1
0.4	80.2	1.6	67.6	5.1	71.4	85.4	76.6	44.
0.5	78.7	1.0	67.0	6.9	75.2	93.8	78.3	17.

Table 20: Grid search for ABL against representative attacks on Mobilenetv2 (Default represents the result under the default setting provided by backdoorbench).

552		BADNET		BLE	NDED	WANET	
533	GAMMA	CA	ASR	CA	ASR	CA	ASR
534	DEFAULT	68.0	24.4	67.3	2.6	50.9	0.5
535	0.0	81.3	36.9	77.3	21.6	68.9	74.8
	0.1	81.3	36.9	77.3	21.6	68.9	74.8
536	0.2	81.3	36.9	77.3	21.6	68.9	74.8
537	0.3	81.3	36.9	77.3	21.6	68.9	74.8
	0.4	79.7	13.6	78.4	15.1	68.9	74.8
538	0.5	81.3	48.1	80.4	33.4	68.9	74.8

Table 21: Result of DBD against additional attacks on ResNet18 (Default represents the result under the default setting provided by backdoorbench, Default2 represents the recommended setting used in ASD).

CETTING	L	С	SIG		Refool		NARCISSUS	
SETTING	CA	ASR	CA	ASR	CA	ASR	CA	ASR
DEFAULT	83.2	98.1	77.6	99.9	87.0	0.1	87.3	99.6
DEFAULT2	82.46	99.42	80.12	99.9	90.84	2.34	80.73	99.61

Table 22: Result of DBD against representative attacks on Mobilenetv2 (Default represents the result under the default setting provided by backdoorbench, Default2 represents the recommended setting used in ASD).

1560							
1561	CETTING.	BADNET		BLENDED		WANET	
1501	SETTING	CA	ASR	CA	ASR	CA	ASR
1562	DEFAULT	65.5	0.0	69.0	0.0	58.4	12.4
1563	DEFAULT2	54.34	0	64.22	0	57.22	14.11
1564				1		1	