DO WE REALLY NEED LABELS FOR BACKDOOR DEFENSE?

Anonymous authors
Paper under double-blind review

ABSTRACT

Since training a model from scratch always requires massive computational resources recently, it has become popular to download pre-trained backbones from third-party platforms and deploy them in various downstream tasks. While providing some convenience, it also introduces potential security risks like backdoor attacks, which lead to target misclassification for any input image with a specifically defined trigger (*i.e.*, backdoored examples). Current backdoor defense methods always rely on clean labeled data, which indicates that safely deploying the pre-trained model in downstream tasks still demands these costly or hard-to-obtain labels. In this paper, we focus on how to purify a backdoored backbone with only unlabeled data. To evoke the backdoor patterns without labels, we propose to leverage the unsupervised contrastive loss to search for backdoors in the feature space. Surprisingly, we find that we can mimic backdoored examples with adversarial examples crafted by contrastive loss, and erase them with adversarial finetuning. Thus, we name our method as Contrastive Backdoor Defense (CBD). Against several backdoored backbones from both supervised and self-supervised learning, extensive experiments demonstrate our unsupervised method achieves comparable or even better defense compared to these supervised backdoor defense methods. Thus, our method allows practitioners to safely deploy pre-trained backbones on downstream tasks without extra labeling costs.

1 INTRODUCTION

While deep neural networks (DNNs) have achieved promising performance on various tasks, including computer vision (He et al., 2016) and natural language processing (Floridi & Chiriatti, 2020), their success heavily relies on a huge amount of data, massive computational resources, and carefully tuning of hyper-parameters. Thus, it becomes popular to download a pre-trained backbone and deploy it on several downstream tasks in recent years (Newell & Deng, 2020; Tan et al., 2018; He et al., 2019). These backbones can be trained in any training paradigms, including supervised learning and self-supervised learning (Chen et al., 2020; He et al., 2022; Gidaris et al., 2018), and then be open-sourced on third-party platforms.

While providing convenience, they also bring potential risks such as backdoor attacks. Numerous works (Gu et al., 2017; Nguyen & Tran, 2021; Turner et al., 2019) pointed out this threat easily occurs in supervised learning, and recent studies (Saha et al., 2022; Jia et al., 2022) started to pay attention to backdoor attacks in self-supervised learning. Specifically, a backdoored DNN always predicts a predefined label for any input image with a specific trigger. For example, a traffic sign recognition system based on a backdoored backbone may always predict the "STOP" sign as "GO STRAIGHT" in the presence of a specific pattern, which causes severe security problems.

To address this security issue, many defense methods (Zeng et al., 2022; Wang et al., 2019; Wu & Wang, 2021) are proposed. Unfortunately, almost all methods focus on backdoor attacks inside supervised learning DNNs by building a classification-based loss to defend. In the popular deployment scheme from the pre-trained backbone to downstream tasks, the practitioners might have few costly labeled data, fail to obtain a classifier head (*e.g.*, a self-supervised backbone) to compare with the true label, or feel difficult to design a classification-based loss (*e.g.*, tasks for detection or segmentation). To break through these restrictions, we first consider the following question:

Do we really need labels for backdoor defense?

In this paper, we focus on how to purify a backdoored backbone with only unlabeled data. Regarding the backdoor trigger as a "shortcut" (Wang et al., 2019) in decision boundary (a small trigger is enough to change outputs for many backdoored models), the traditional methods (Wang et al., 2019; Zeng et al., 2022) attempt to make the prediction deviate from the ground-truth label as far as possible using a small perturbation in inputs, so as to evoke the backdoor behavior and then erase it. Unfortunately, we have no access to any labels, and even the prediction results if the backbone lacks a classifier head. To evoke the backdoor behavior without labels, we propose to leverage the unsupervised contrastive loss to search for the backdoor in the feature space, *i.e.*, we try to make the output feature as different from its original feature as possible using a small perturbation. Surprisingly, we find that we can easily mimic backdoored examples with adversarial examples crafted by contrastive loss. Based on this finding, we propose to erase the backdoor behaviors by letting these contrastive loss-based adversarial have similar features as their clean counterpart using fine-tuning. Thus, we term our method as Contrastive Backdoor Defense (CBD), which successfully defends against backdoor attacks without any labeled data. Our main contributions are summarized as follows,

- We explore a more practical backdoor defense that requires no access to labeled data or the classifier head. It is quite suitable in the recently popular case in which the practitioner downloads a pre-trained backbone and then deploys it in the downstream tasks.
- We find that adversarial samples generated by the contrastive loss approach the cluster of backdoor samples in the hidden feature space. Inspired by it, we introduce a fine-tuning based method that can purify the backdoored backbone without any labeled data.
- We conduct comprehensive experiments to verify the effectiveness of our method across different datasets and backdoor attacks. Empirically, our unsupervised method achieves comparable or even better defense compared to previous supervised defense.

2 Related work

Backdoor Attack. Backdoor Attack is a newly risen security concern on DNNs(Gu et al., 2017), in which the adversary can manipulate the model to predict a target class as long as a predefined trigger pattern appears in the image. This backdoor behavior can be easily injected inside DNNs by poisoning some data pairs. Specifically, (1) **poison-label attack**: the attacker randomly adds the trigger pattern into samples from all classes and changes their label to the target class (Gu et al., 2017; Chen et al., 2017; Nguyen & Tran, 2021; Zbontar et al., 2021; Doan et al., 2021)). (2) **clean-label attack**: the adversary only adds the trigger pattern into the samples from the target class, which is more stealthy since their annotation is correct (Turner et al., 2019). Recent studies start to pay attention to backdoor attacks on self-supervised learning frameworks, especially on contrastive learning methods(Saha et al., 2022; Jia et al., 2022). This emerging threat is challenging for DNN models and attracts researchers' attention.

Backdoor Defense. Meanwhile, numerous defense methods are proposed, which can be mainly grouped into two categories, including (1) **training-time defense**(Li et al., 2021a; Huang et al., 2022; Gao et al., 2021): the defender can access training data and train a model based on various defense strategies. For instance, Gao et al. (2021) utilizes adversarial training to train a robust model against backdoor triggers; (2) **post-processing defense**(Liu et al., 2018; Wang et al., 2019; Wu & Wang, 2021; Zeng et al., 2022; Li et al., 2021b): the defender sanitize the models with tiny amounts of data with no access to the training process and training data. Thus, post-processing defense can be applied in a wider range of scenarios, *e.g.*, purifying backbones from the Internet before deploying them in downstream tasks. However, almost all these methods rely on enough amount of labeled clean data and classification loss, while labeled data may be hard to obtain, the backbone may have no classifier head, or it is hard to design classification-based loss for defense (*e.g.*, defense for detection or detection). In this work, we focus on how to purify backdoored backbone from both supervised learning and self-supervised learning.

Self-supervised Learning. Self-supervised learning methods rely on a pretext task(*e.g.*, rotations predictions, jigsaw puzzles) to perform pre-training steps and use the rich pre-trained representations to deploy various downstream tasks such as object detection. In recent years, contrastive

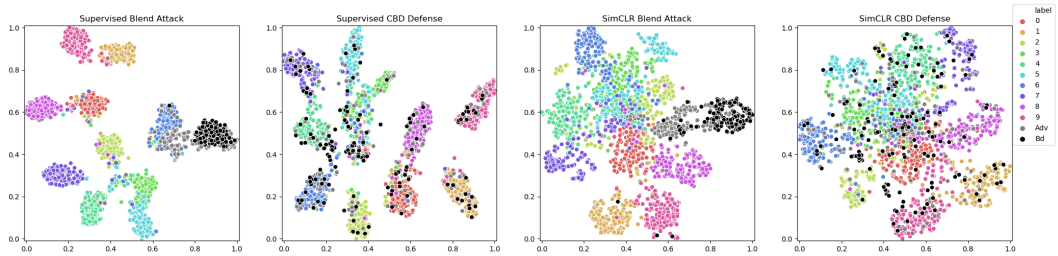


Figure 1: The t-SNE visualization of samples in feature space on CIFAR-10. All backbones are backdoored by the Blend attack with class 6 as the target label. *Bd*: backdoor examples; *Adv*: contrastive adversarial examples.

learning(CL) has demonstrated superior performance to learn more generalizable features in a wide range of vision tasks(He et al., 2020; Zbontar et al., 2021; Chen et al., 2020).

Adversarial Attack. Adversarial attacks are inference-time attacks that aim to mislead the DNNs. There are various attack methods(Goodfellow et al., 2014; Moosavi-Dezfooli et al., 2017; Xiao et al., 2018). In this work, we mainly leverage the typical untargeted PGD attack(Madry et al., 2018) to generate adversarial examples on backdoored DNN. Untargeted PGD attack finds adversarial examples iteratively by maximizing the loss of the samples and their labels.

3 Contrastive Backdoor Defense without Any Labeled Data

In this section, we first define our problem setup. Then, we analyze how to discover potential backdoored features with unsupervised contrastive loss. Finally, based on this observation, we propose a novel fine-tuning method for a pre-trained feature extractor.

3.1 Preliminaries

Backbone Training. For the backbone $f(\cdot, \theta)$ from supervised learning, its parameters are usually trained based on a K-class classification problem. Given a labeled training dataset $\mathcal{D}_l = \{(x_1, y_1), \cdots, (x_N, y_N)\}$, which contains N inputs $x_i \in \mathbb{R}^d, i = 1, \cdots, N$, and the corresponding ground-truth label $y_i \in \{1, \cdots, K\}$, the cross-entropy loss for a single data pair (x_i, y_i) can be calculated as follows,

$$\ell_{xe}(x_i) = -\log g_{y_i}(f(x_i, \theta)), \tag{1}$$

where the $g(\cdot)$ is the classifier head for this classification task and $g_{y_i}(\cdot)$ indicates the outputted probability that x_i belongs to class y_i from the classifier. The training process attempts to find an optimal model parameter θ to minimize the average loss on the whole training data,

By contrast, the backbone $f(\cdot, \theta)$ from self-supervised learning is trained without any classifier head. The training process optimized the model parameters, so as to let the similar sample pairs stay close to each other while dissimilar ones are far apart in the embedding space. For example, given an unlabeled dataset $D_u = \{x_1, \cdots, x_N\}$, the normalized temperature-scaled cross-entropy loss (NT-Xent loss) for the sample x_i is as follows,

$$\ell_{cl}(\tilde{x}_i, \hat{x}_i) = -\log \frac{\exp(\text{sim}(\tilde{z}_i, \hat{z}_i)/\tau)}{\sum_{k=1}^{2N} \mathbb{I}_{[k \neq i]} \exp(\text{sim}(\tilde{z}_i, \tilde{z}_k)/\tau)}, \tag{2}$$

where $\tilde{z}_i = f(\tilde{x}_i, \theta), \hat{z}_i = f(\hat{x}_i, \theta), \tilde{z}_k = f(\tilde{x}_k, \theta)$, $\text{sim}(\cdot, \cdot)$ is the cosine similarity and τ is the temperature hyper-parameter. $\tilde{x}_i$ and $\hat{x}_i$ (positive samples) are two augmented samples from the same sample x_i, while $\hat{z_k}$ (negative samples) is the projection of any other augmented sample in the whole dataset[1]. Then, the self-supervised training process attempts to find an optimal model parameter θ to minimize the average loss over all possible positive pairs on the unlabeled dataset.

[1]To reduce the computational burden, we usually project the feature $f(\cdot, \theta)$ into a small space before computing the NT-Xent loss, and compute the loss within a mini-batch rather than the whole dataset

Backdooring the Backbone. It is easy to inject the backdoor behavior into the supervised learning model. Specifically, the adversary adds a trigger pattern Δ (*e.g.*, a 3×3 checkerboard) into the images x_i to obtain a triggered sample x_i', and then relabel it as a predefined target label y_i'. The adversary could replace some training data $\{(x_1, y_1), \cdots, (x_M, y_M)\} \subset D_l$ with their poisoned counterparts $\{(x_1', y_1'), \cdots, (x_M', y_M')\}$ before training. If the model successfully fits these data, they are embedded by the backdoor behavior.

However, it is more difficult to inject backdoor behavior into a self-supervised model since the adversary is unable to manipulate the label. Instead, Saha et al. (2022) proposed to add the trigger pattern into samples from the target class.

Threat Model. In this paper, we assume the practitioner downloads a pre-trained backbone, and defends against potential backdoor attacks before deploying it. In particular, the defense requires no access to the classifier and labeled data. This is a more general setting than previous settings, which covers much more situations: 1) the backbone has no classifier head (*e.g.*, self-supervision backbone); 2) the labeled data is too costly to collect enough (e.g., medical application); 3) it is difficult to design a classification-based loss (*e.g.*, tasks for detection or segmentation). We hope the final algorithm can serve as an off-the-shelf solution that is not limited to supervised learning and is able to handle various situations, so as to allow practitioners to safely and painlessly deploy pre-trained backbones on downstream tasks.

3.2 Observations in the hidden feature space

In this section, we first present our observation of backdoor attacks in contrastive model, then analyze the motivations to connect poisoned samples and instance-wise adversarial examples. Furthermore, we show how to design an adversarial attack without labels.

Observations in The Hidden Feature Space. To dive into the backdoored backbone from both supervised and self-supervised learning, we illustrate the representation of poisoned samples in the hidden feature space. We start with the supervised backbone, where the clean samples from the target class (blue circles) and the backdoor samples (black circles) are located in two separate clusters respectively as shown in the 1*st* plot in Figure 1, which is a similar finding as in Huang et al. (2022). As for the self-supervised backbone, we also find the backdoor cluster is obviously separate from the clean cluster in the 3*nd* plot in Figure 1, which is different from Huang et al. (2022). Overall, the backdoor cluster is always separate from the clean cluster belonging to the same label on both supervised learning and self-supervised learning. Note that, since the backdoor cluster and clean clusters of the target class are separate, we have a chance to suppress backdoor behaviors via overlapping these clusters if we could obtain these backdoor samples and the target class.

Adversarial Samples and Backdoor Samples. Remember we never know the specific type of backdoor behavior injected in the backbone, and also have no access to the knowledge of backdoor samples and their corresponding target class. Previous research (Wang et al., 2019) regards the backdoor trigger as a "shortcut" in decision boundary, because a small trigger is enough to change outputs for many backdoored models. This provides a possible approach, *i.e.*, making the prediction of the backdoor sample deviate from its ground-truth label as far as possible using a small perturbation in inputs, so as to reconstruct this "shortcut". Unfortunately, this method requires labeled data, which might be hard to obtain or costly[2]. Inspired by contrastive adversarial examples that only depends on unlabeled instances(Fan et al., 2021; Kim et al., 2020; Jiang et al., 2020), we attempt to make the contrastive loss between two views of a benign instance as far as possible in the hidden feature space.

CL-based Adversarial Example Generation. Generating untargeted adversarial examples on feature space is also important. Although Saha et al. (2022) suggests that adversarial training can not be straightforwardly applied to the SSL paradigm as the lack of labels, recent works (Jiang et al., 2020; Kim et al., 2020; Fan et al., 2021) propose novel ways to generate adversarial examples on contrasitve loss. Inspired by Jiang et al. (2020), we use the single image perturbation method to generate the potential backdoored instances. We only add perturbation on one of two augmentations in contrastive loss(ℓ_{cl}). Suppose we have an image x and a pair of augmentations $(\tilde{x}_i, \tilde{x}_j)$. To generate an potential backdoored instance of image x, we want to maximize the contrastive loss

[2]This is also the reason why the practitioners prefer to deploy a pre-trained model in downstream tasks.

between features of augmentation $\tilde{x_j}$ and augmentation $\tilde{x}_i + \delta_i$ from backbone $f(\cdot, \theta)$ using PGD attack(Madry et al., 2018), where δ_i is the perturbation. Formally, perturbation δ_i can be gotten by iteratively optimizing:

$$\max_{\|\delta_k\|_p \leq \epsilon} \ell_{cl}(\tilde{x_i} + \delta_i, \tilde{x_j})$$

where $\|.\|_p$ is L_p norm and ϵ is perturbation budget. Note that in our method, we do not use adversarial training. Instead, only one-time generations are required.

3.3 The Proposed Method

Inspired by all these findings, we propose a fine-tuning-based defense method that can erase the backdoor without any requirements for labels. We propose a Backdoor-to-Standard loss based on the above findings, leverage the power of standard contrastive fine-tuning to align the clean samples, and finally add a contrastive loss based knowledge distillation, where the original model will act as a teacher and the current as a student.

Backdoor-to-Standard Pulling. We first illustrate how to mitigate backdoor-sensitive neurons. In supervised learning, the backdoor attack mainly builds a strong connection between the trigger and its target class(Huang et al., 2022). Thus, connecting the trigger to all the classes can effectively break the backdoor attack. However, this approach is infeasible in our setting since we do not use labeled data. Instead, we design a feature space alignment of backdoored neurons and clean neurons. Based on our backdoored feature generation in the last section, we suspect those generated instances may act similar neurons as those of triggered images. Thus, pulling adversarial images toward its benign parts can mitigate backdoor effects as the active neurons of backdoor is changed. Specifically, for each image x, in addition with $(\tilde{x_i}, \tilde{x_j})$, we want to generate another view $\tilde{x_k}$, and find its perturbation δ_k by using augmentation pair $(\tilde{x_i}, \tilde{x_k})$. After obtaining $\tilde{x_k} + \delta_k$, we treat it as a trigger recovered image, and our Backdoor-to-Standard pulling loss will be:

$$\ell_{pull}(x) = \ell_{cl}(freeze(\tilde{x_i}), \tilde{x_k} + \delta_k).$$

It is worth noting that we freeze the feature of $\tilde{x_i}$. Since we do not want to pull features of $\tilde{x_i}$ close to those of triggered images, we treat it as an anchor.

Standard Fine-tuning. Then, to retain the benign performance, standard contrastive fine-tuning between benign augmentations on trained backbones is crucial. We follow the same manner as normal fine-tuning does in supervised learning. However, instead of fine-tuning with cross-entropy loss and a linear classifier, we directly utilize the contrastive loss on backbone. Therefore, our standard fine-tuning will be:

$$\ell_{sft}(x) = \ell_{cl}(\tilde{x_i}, \tilde{x_j}),$$

For standard fine-tuning, we want to preserve the clean performance of our model.

Embedding Distillation. With our discussion in the last section and the clean performance of backdoored backbone, we suspect that the poisoned model will still maintain good feature distributions except for the poisoned sample. In the meantime of suppressing backdoor-sensitive neurons, the clean samples can also share the distribution of the original backbone. Thus, for the last part of our method, we want the purified model to have similar clean behaviors to the original poisoned model. In order to do this, knowledge distillation is a straightforward option. We are using poisoned model as a teacher, and our current model as a student. Tian et al. (2020) proved that the idea of contrastive loss can be well applied to knowledge distillation in embedding space. Based on this idea, we want to align the clean features from fine-tuned model to our original model:

$$\ell_{kd}(x) = \ell_{cl}(\tilde{x_i}, (\tilde{x_i}, \theta^*)).$$

θ^* here is the parameter of the poisoned model, we get the feature of $\tilde{x_i}$ from $f(\tilde{x_i}, \theta^*)$. It is important to note that this distillation will not preserve the backdoored feature distribution, as we assumed all of our fine-tuning data are clean.

Overall, the loss on sample x of our fine-tuning method is a combination of three items:

$$\ell_{total} = \lambda_1 \ell_{pull}(x) + \lambda_2 \ell_{sft}(x) + \lambda_3 \ell_{kd}(x),$$

where λ_1, λ_2, and λ_3 are positive hyper-parameters and $\lambda_1 + \lambda_2 + \lambda_3 = 1$.

4 EXPERIMENTS

Datasets and DNNs. We evaluate the performance of our method on **Cifar-10 dataset**(Krizhevsky et al., 2009). We use Resnet-18(He et al., 2016) as the backbone for both supervised and CL model and select SimCLR(Chen et al., 2020) as our CL method. We train 200 epochs for supervised model and 1000 epochs for SimCLR.

Backdoor Attack Configuration. We use BadNets(Gu et al., 2017), Blend(Chen et al., 2017), SIG(Barni et al., 2019), and WaNet(Nguyen & Tran, 2021) as our attack baselines. BadNet is a patch-based attack; we apply a 3×3 checkerboard as our trigger. For supervised setting, we put it on the top-left. For self-supervised setting, we put it in the center of the image. Blend is a global attack; we generate a 32×32 Gaussian noise image and set the blend ratio $\alpha = 0.2$ for Blend. For SIG, we adopt poison-label setting and set the $f = 6$ and $\Delta = 20$. We use the official code and configurations except for the poisoning ratio of WaNet. However, we only show the results of WaNet under supervised learning since it can not effectively create backdoor during our experiments on contrastive learning. To achieve better attack results on contrastive learning model, we set the target label to 6. The poisoning ratio across all supervised settings is 6% of the dataset, and the noise ratio for WaNet is set to 2 as default. To create backdoor on self-supervised model, we poison 60% of target label 6, which is 6% of all data. More details of the configurations can be found in Appendix A.1.

Backdoor Defense Configuration. For supervised setting, we select fine-tuning and state-of-the-arts model post-processing defenses including ANP(Wu & Wang, 2021), NAD(Li et al., 2021b), and Fine-pruning(Liu et al., 2018) as our baseline methods. ANP is a pruning-based method; we use the default configuration for ANP. When evaluating ANP, we make a trade-off between clean accuracy and ASR. Fine-tuning, NAD, and Fine-pruning are all fine-tuning-based methods. We adopt the settings for these defenses from BackdoorBenchmark(Wu et al., 2022). Specifically, we fine-tune the backdoored models for 10 epochs. We also use this fine-tuned model as the teacher model for NAD and run the NAD defense for 20 epochs. In addition, we prune the Fine-pruning defense until the clean accuracy reaches 90% tolerance ratio. We then fine-tune the pruned model with 50 epochs.

For self-supervised setting, we only consider standard contrastive fine-tuning as our baseline. Standard contrastive fine-tuning is the special case of our method when hyper-parameter is $\lambda_1 = 0$, $\lambda_2 = 1$, and $\lambda_3 = 0$.

For our CBD, we set the default hyper-parameters to $\lambda_1 = 0.3$, $\lambda_2 = 0.2$, and $\lambda_3 = 0.5$ for both training settings. We generate adversarial examples with $\epsilon = 8$, step sizes $= 0.1$, and step $= 20$ for supervised learning and step $= 100$ for self-supervised learning. We run CBD for 15 epochs. More details of defense configurations can be found in Appendix A.2.

Evaluation metric. To evaluate the performance of clean examples, we use the clean accuracy (Acc). For the evaluation of poisoned samples, we use two different methods. The first method is the standard attack success rate (ASR) which measures the false positive samples of the targeted class. In addition, we consider the patched accuracy (PA) which denotes the original accuracy of triggered samples as a more comprehensive demonstration of the attack. The intuition behind this measurement is some triggered samples are not classified as the targeted class despite the fact that they get poor ASR. This phenomenon occurs in self-supervised learning. We believe this is because some triggers, especially global triggers, can only ensure misleading features(Carlini & Terzis, 2022). Thus, only measuring ASR for this type of attack is insufficient. When evaluating self-supervised experiments and our CBD defense, we train a linear classifier on top of the pre-trained features with 1% of clean train data.

4.1 EFFECTIVENESS OF CBD DEFENSE

The effectiveness of the proposed CBD method is shown in this section. We use 5% of unlabeled data for CBD across different experiments.

Comparing labeled data. We first compare CBD with four post-processing methods under supervised setting. We assume all the baseline methods can access the classifier head for better comparison. For the trade-off of getting costly labeled data, we compare these methods on 1% extra clean labeled data. We demonstrate the results in Table 1. Our method successfully purified different

Attacks	Metrics	Before	ANP	FP	FT	NAD	CBD (ours)
BadNets	ACC	93.25	91.34	91.11	**91.52**	90.76	89.74
	ASR	99.95	0.10	1.72	7.59	1.58	**1.07**
	PA	0.06	89.77	88.89	85.59	89.19	**89.34**
Blend	ACC	94.23	88.94	91.54	**92.38**	89.32	91.81
	ASR	100	37.41	54.02	98.89	72.97	**4.98**
	PA	0	38.83	32.31	1.07	20.26	**81.33**
SIG	ACC	94.45	89.43	91.09	**92.26**	89.28	90.89
	ASR	99.29	2.08	4.38	5.24	**2.97**	4.98
	PA	0.67	82.57	80.59	**81.11**	78.62	78.88
WaNet	ACC	93.67	93.41	**93.37**	91.99	88.88	88.81
	ASR	94.88	0.99	0.28	7.67	**1.01**	3.82
	PA	4.94	92.56	91.49	83.64	**87.08**	85.98

Table 1: Results on supervised Cifar-10. Accuracy(Acc) of the clean test data, Attack Sucess Rate(ASR) on poisoned test data with target labels, and Patched Accuracy(PA) on poisoned test data with original labels. The best results of fine-tuning-based methods are in **bold**.

Attacks	Metrics	Before	ANP	FP	FT	NAD	CBD (ours)
BadNets	ACC	93.25	91.54	92.54	92.71	**92.94**	89.74
	ASR	99.95	0.08	4.12	11.33	6.20	**1.07**
	PA	0.06	87.49	**89.51**	83.98	88.26	89.34
Blend	ACC	94.23	87.92	93.11	**93.47**	93.43	91.81
	ASR	100	48.69	75.30	60.73	66.72	**4.98**
	PA	0	31.07	20.84	27.92	24.01	**81.33**
SIG	ACC	94.45	90.66	**93.13**	92.55	92.73	90.89
	ASR	99.29	2.10	10.46	72.44	57.53	**4.98**
	PA	0.67	84.98	77.31	25.78	39.91	**78.88**
WaNet	ACC	93.67	92.03	**93.59**	89.94	91.57	88.81
	ASR	94.88	0.69	**0.19**	0.96	0.69	3.82
	PA	4.94	90.67	**92.49**	88.18	89.86	85.98

Table 2: Results for pseudo-label data.

Defenses	BadNets			Blend			SIG		
	Acc	ASR	PA	Acc	ASR	PA	Acc	ASR	PA
No Defense	85.68	28.73	61.10	85.36	43.01	23.12	85.43	33.34	56.97
FT	64.94	5.20	53.32	63.83	3.79	20.93	66.59	7.97	47.93
Our method	81.77	6.70	76.20	80.07	3.67	70.69	80.51	6.64	66.34

Table 3: Results for SimCLR, FT here is contrastive fine-tuning.

attacks even without labeled data and classifier head. The attack success rate in all cases is less than 5%. Also, the gap between the original clean performance and our purified clean performance is less than 5%. More specifically, on Blend attack, we are the only method that effectively defends the backdoor. On SIG and BadNets, we reach comparable results with other baselines. We achieved the lowest ASR across all fine-tuning-based methods against BadNets and even higher clean accuracy than state-of-the-art ANP and NAD on SIG. CBD still generates acceptable performance in the worst case of WaNet. However, we can adjust the weight of losses or adversarial strategies to improve its performance.

Comparing pseudo-label data We are comparing CBD with other baselines on 5% unlabeled data with generated pseudo labels in Table 2. Note that this practice is only applicable when the classifier head is retained and clean accuracy is high (*e.g.*, Cifar-10). Even though the generated pseudo labels are mostly accurate, the existing noise of these labels deteriorates other defense baselines. In particular, Fine-tuning and NAD lost their ability to defend SIG, and the ASR of fine-tuning-based methods in most cases is increased. Moreover, despite comparable results on SIG for ANP, the

Defenses	ImageNet-100			VOC2007	
	Acc	ASR	PA	AP50	PAP50
No Defense	59.94	2.7	40.59	64.98	38.46
Our method	48.06	1.06	43.76	64.08	44.77

Table 4: Case study for object detection. AP is the evaluation metric for object detection. Higher AP is better. PAP is the patched AP for our backdoor evaluation.

Attacks	Metrics	Before	$\epsilon = 2$	$\epsilon = 4$	$\epsilon = 16$	$L_2 1024$	PGD100	CBD (ours)
BadNets	ACC	93.25	89.86	89.72	89.82	**92.16**	88.62	89.74
	ASR	99.95	1.14	1.24	**1.06**	4.65	1.22	1.07
	PA	0.06	89.38	89.12	**89.48**	88.73	88.19	89.34
Blend	ACC	94.23	91.22	91.16	92.21	**93.03**	90.99	91.81
	ASR	100	10.76	8.50	3.35	**3.12**	4.91	4.98
	PA	0	73.93	76.13	**84.29**	82.43	81.90	81.33
SIG	ACC	94.45	91.09	90.79	90.98	**92.96**	89.51	90.89
	ASR	99.29	5.01	**4.49**	5.58	43.54	4.95	4.98
	PA	0.67	78.29	**78.90**	78.46	51.83	77.73	78.88
WaNet	ACC	93.67	89.20	88.33	88.25	**90.51**	86.57	88.81
	ASR	94.88	1.17	1.22	1.65	**1.15**	2.02	3.82
	PA	4.94	88.93	87.99	87.66	**89.90**	85.60	85.98

Table 5: Ablation studies with different adversarial example strategies on supervised backbones.

patched accuracy on other attacks is decreased. These results demonstrate the weakness of using pseudo-labels to defend against backdoor attack.

Comparing self-supervised learning Table 3 shows the results on self-supervised backbone. We pick SimCLR as our self-supervised learning method. It is worth noting that self-supervised methods in the real-world only leverage unlabeled data. Thus, we only demonstrate the effectiveness of our method and use contrastive fine-tuning as a comparison. All the performances of backbones are tested under a linear classifier with 1% training data. While maintaining high clean accuracy, CBD successfully defends all attacks and recovers the patched accuracy. Compared with standard contrastive fine-tuning that greatly downgrades the clean performance and achieves low patched accuracy, we verify the effectiveness of CBD under contrastive learning. More results for self-supervised backbones can be found in Appendix C. Also, we provide a case study for ImageNet-100(Tian et al., 2020; Deng et al., 2009) and object detection in Table 4. In this case study, we demonstrate the effects of backdoor attack on the backbone and improve the patched accuracy and patched AP with our CBD. We train the backbone model using Blend attack with 224×224 Gaussian noise on ImageNet-100. Then, we train and test the Fast-RCNN with this backbone on PascalVOC-2007(Everingham et al., 2009) dataset. We run our CBD with only 1% of the ImageNet-100 dataset. Even in the downstream object detection setting, our method increases the patched AP scores by more than 6%. These results pointed out the emergent threat of pre-trained backbone and verify the effectiveness of our method. More implementation details for ImageNet-100 and object detection are in Appendix B.

4.2 Ablation Studies

In this section, we present ablation studies on supervised backbones. More results on self-supervised backbones can be found in Appendix C

Effects on different adversarial examples The quality of adversarial examples that mimic backdoored features plays an important role in our method. Among different adversarial settings, PGD20 attack with L_∞ norm, 0.1 step size, and $\epsilon = 8$ is the default in our supervised setting. Please note that our default settings are not the best; we have made some trade-offs. As shown in Table 5, we present the results with different budgets($\epsilon = 2, 4, 16$), steps(PGD100), and norm(L_2 norm with $\epsilon = 1024$). The carefully selected results can reach state-of-the-art without any labeled data.

Attacks	Metrics	Before	FT	w/o FT	w/o KD	Pulling	CBD (ours)
BadNets	ACC	93.25	88.55	88.84	80.65	71.76	**89.74**
	ASR	99.95	12.01	1.18	3.23	4.18	**1.07**
	PA	0.06	77.83	88.59	80.08	71.28	**89.34**
Blend	ACC	94.23	89.21	91.03	85.68	75.70	**91.81**
	ASR	100	91.89	5.18	5.59	6.77	**4.98**
	PA	0	6.99	81.10	76.34	67.34	**81.33**
SIG	ACC	94.45	89.38	89.78	84.12	74.39	**90.89**
	ASR	99.29	14.00	5.22	5.50	7.32	**4.98**
	PA	0.67	69.54	78.40	70.61	59.13	**78.88**
WaNet	ACC	93.67	69.80	86.42	59.81	54.90	**88.81**
	ASR	94.88	3.91	**1.66**	6.20	5.78	3.82
	PA	4.94	69.76	**86.37**	58.72	51.70	85.98

Table 6: Ablation studies of different losses on supervised backbones. FT here is contrastive fine-tuning and Pulling is Backdoor-to-Standard pulling.

Based on the results of these adversarial examples, they successfully reach backdoored features with instance-wise contrastive loss.

Effects on different losses. Our proposed CBD is composed of three losses. To verify the effectiveness of these losses, we test our methods on four different variants. Specifically, we adjust the hyper-parameters of these losses accordingly. These variants include (1) contrastive fine-tuning($\lambda_2 = 1$), (2) CBD without contrastive fine-tuning ($\lambda_1 = 0.5, \lambda_3 = 0.5$), (3) CBD without knowledge distillation($\lambda_1 = 0.5, \lambda_2 = 0.5$), and (4) Backdoor-to-Standard pulling($\lambda_1 = 1$). We can verify their contributions to our method on Table 6. In most cases, our default setting reaches the best performance. This indicates that all three losses make important contributions in real practice.

5 Conclusion

In this paper, we proposed a fine-tuning-based method to erase the backdoor behaviors inside pre-trained backbones. We analyzed the behaviors of backdoored examples and proposed a contrastive loss-based way to approach them, based on which, we propose the Contrastive Backdoor Defense (CBD) to remove the backdoors without needing any labeled data. Extensive experiments on different datasets with various attacks demonstrate the superiority of our proposed method on different experimental settings with only unlabeled data.

References

Mauro Barni, Kassem Kallas, and Benedetta Tondi. A new backdoor attack in cnns by training set corruption without label poisoning. In *ICIP*, 2019.

Nicholas Carlini and Andreas Terzis. Poisoning and backdooring contrastive learning. In *ICLR*, 2022.

Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. In *ICML*, 2020.

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

Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *CVPR*, 2009.

Khoa Doan, Yingjie Lao, Weijie Zhao, and Ping Li. Lira: Learnable, imperceptible and robust backdoor attacks. In *ICCV*, 2021.

Mark Everingham, Luc Van Gool, Christopher K. I. Williams, John M. Winn, and Andrew Zisserman. The pascal visual object classes (voc) challenge. *International Journal of Computer Vision*, 88:303–338, 2009.

Lijie Fan, Sijia Liu, Pin-Yu Chen, Gaoyuan Zhang, and Chuang Gan. When does contrastive learning preserve adversarial robustness from pretraining to finetuning? In *NeurIPS*, 2021.

Luciano Floridi and Massimo Chiriatti. Gpt-3: Its nature, scope, limits, and consequences. *Minds and Machines*, 30(4):681–694, 2020.

Yinghua Gao, Dongxian Wu, Jingfeng Zhang, Shu-Tao Xia, Gang Niu, and Masashi Sugiyama. Does adversarial robustness really imply backdoor vulnerability? 2021.

Spyros Gidaris, Praveer Singh, and Nikos Komodakis. Unsupervised representation learning by predicting image rotations. In *ICLR*, 2018.

Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572*, 2014.

Tianyu Gu, Brendan Dolan-Gavitt, and Siddharth Garg. Badnets: Identifying vulnerabilities in the machine learning model supply chain. *arXiv preprint arXiv:1708.06733*, 2017.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, 2016.

Kaiming He, Ross Girshick, and Piotr Dollár. Rethinking imagenet pre-training. In *ICCV*, 2019.

Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross B. Girshick. Momentum contrast for unsupervised visual representation learning. In *CVPR*, 2020.

Kaiming He, Xinlei Chen, Saining Xie, Yanghao Li, Piotr Dollár, and Ross Girshick. Masked autoencoders are scalable vision learners. In *CVPR*, 2022.

Kunzhe Huang, Yiming Li, Baoyuan Wu, Zhan Qin, and Kui Ren. Backdoor defense via decoupling the training process. In *ICLR*, 2022.

Jinyuan Jia, Yupei Liu, and Neil Zhenqiang Gong. Badencoder: Backdoor attacks to pre-trained encoders in self-supervised learning. In *IEEE S&P*, 2022.

Ziyu Jiang, Tianlong Chen, Ting Chen, and Zhangyang Wang. Robust pre-training by adversarial contrastive learning. In *NeurIPS*, 2020.

Minseon Kim, Jihoon Tack, and Sung Ju Hwang. Adversarial self-supervised contrastive learning. In *NeurIPS*, 2020.

Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009.

Yige Li, Xixiang Lyu, Nodens Koren, Lingjuan Lyu, Bo Li, and Xingjun Ma. Anti-backdoor learning: Training clean models on poisoned data. In *NeurIPS*, 2021a.

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

Kang Liu, Brendan Dolan-Gavitt, and Siddharth Garg. Fine-pruning: Defending against backdooring attacks on deep neural networks. In *RAID*, 2018.

Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks. In *ICLR*, 2018.

Seyed-Mohsen Moosavi-Dezfooli, Alhussein Fawzi, Omar Fawzi, and Pascal Frossard. Universal adversarial perturbations. In *CVPR*, 2017.

Alejandro Newell and Jia Deng. How useful is self-supervised pretraining for visual tasks? In *CVPR*, 2020.

Tuan Anh Nguyen and Anh Tuan Tran. Wanet - imperceptible warping-based backdoor attack. In *ICLR*, 2021.

Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In *NeurIPS*, 2015.

Aniruddha Saha, Ajinkya Tejankar, Soroush Abbasi Koohpayegani, and Hamed Pirsiavash. Backdoor attacks on self-supervised learning. In *CVPR*, 2022.

Chuanqi Tan, Fuchun Sun, Tao Kong, Wenchang Zhang, Chao Yang, and Chunfang Liu. A survey on deep transfer learning. In *International conference on artificial neural networks*, pp. 270–279. Springer, 2018.

Yonglong Tian, Dilip Krishnan, and Phillip Isola. Contrastive representation distillation. In *ICLR*, 2020.

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

Bolun Wang, Yuanshun Yao, Shawn Shan, Huiying Li, Bimal Viswanath, Haitao Zheng, and Ben Y. Zhao. Neural cleanse: Identifying and mitigating backdoor attacks in neural networks. In *IEEE S&P*, 2019.

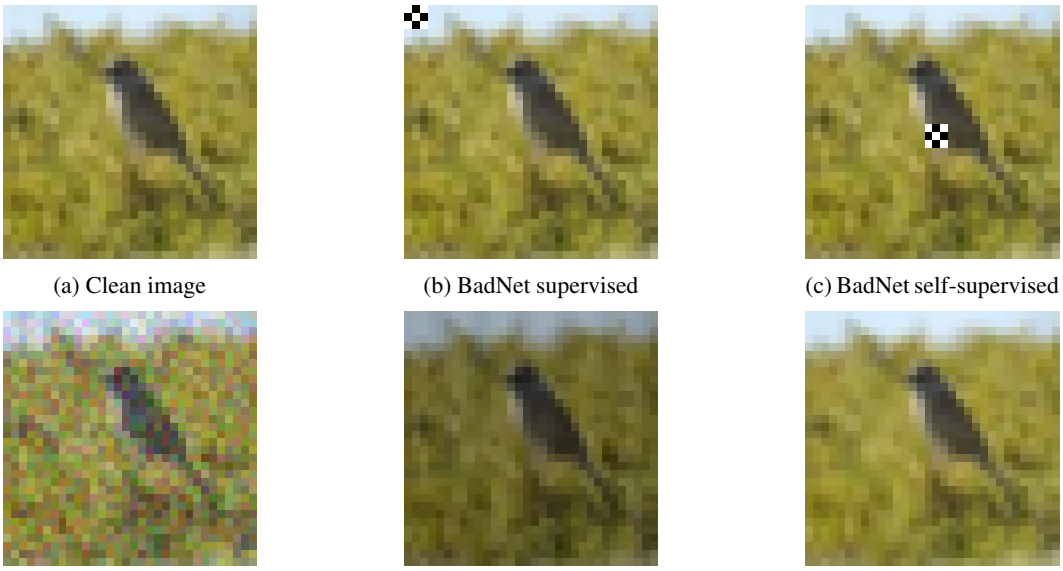
Baoyuan Wu, Hongrui Chen, Mingda Zhang, Zihao Zhu, Shaokui Wei, Danni Yuan, Chao Shen, and Hongyuan Zha. Backdoorbench: A comprehensive benchmark of backdoor learning. In *NeurIPS 2022 Track Datasets and Benchmarks*, 2022.

Dongxian Wu and Yisen Wang. Adversarial neuron pruning purifies backdoored deep models. In *NeurIPS*, 2021.

Chaowei Xiao, Jun-Yan Zhu, Bo Li, Warren He, Mingyan Liu, and Dawn Song. Spatially transformed adversarial examples. In *ICLR*, 2018.

Jure Zbontar, Li Jing, Ishan Misra, Yann LeCun, and Stéphane Deny. Barlow twins: Self-supervised learning via redundancy reduction. In *ICML*, 2021.

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



(a) Clean image (b) BadNet supervised (c) BadNet self-supervised

(d) Blend (e) SIG (f) WaNet

Figure 2: Examples of backdoored images.

A More Implementation details for Backdoor Attacks and Defenses

A.1 Backdoor attacks implementations

In this part, we present the detailed configurations of our attacks. The backdoor triggers are presented in 2. We select label 6 as the target label. We perform poison-label attack with 6% training data for supervised backbone and clean-label attack with 60% of the target category(6% of all data) for self-supervised backbone.

- **BadNets (Gu et al., 2017)**: We implant a 3×3 black-white patch as the trigger. For supervised backbone, we put the trigger on the top-left. We inject it into the center of the image to achieve a better attack success rate on self-supervised backbone.
- **Blend (Chen et al., 2017)**: We mix the data with a 32×32 Gaussian noise image. We achieve good attack results in supervised and self-supervised settings with blend ratio $\alpha = 0.2$. In addition, Blend attack can result in bad patched accuracy in self-supervised backbone even with a low attack success rate. Note that general Gaussian blur augmentation is not presented when training Blend with SimCLR.
- **SIG (Barni et al., 2019)**: We generate SIG trigger with $f = 6$ and $\Delta = 20$. We blend it to target image with blend ratio $\alpha = 0.3$. Note that we use poison-label attack setting for SIG in supervised learning.
- **WaNet (Nguyen & Tran, 2021)**: We use the default configurations and code of WaNet except for the poisoning ratio in supervised backbone. In particular, we poison 6% of training data and set the noise rate $p_n = 2$, $s = 0.5$, and $k = 4$. We implement WaNet based on the original code for self-supervised learning. However, we did not show the results of WaNet on self-supervised backbone as it can not build an effective backdoor with or without noise mode.

A.2 Backdoor Defense implementations

To compare our results with state-of-the-art defense methods, we modify the code from open-source BackdoorBenchmark(Wu et al., 2022). To ensure the fairness of the comparison, we use 1% of

Attacks	Metrics	Before	$\epsilon = 2$	$\epsilon = 4$	$\epsilon = 8$	$\epsilon = 16$	$L_2 1024$	CBD (ours)
BadNets	ACC	85.68	81.69	81.91	**81.99**	81.90	81.93	81.77
	ASR	28.73	14.38	12.42	10.63	**4.01**	8.46	6.70
	PA	61.10	69.32	71.30	72.84	**78.16**	75.21	76.20
Blend	ACC	85.36	79.45	79.69	80.25	80.10	79.97	**80.7**
	ASR	43.01	17.27	12.58	5.26	2.74	**2.23**	3.67
	PA	23.12	49.67	57.82	68.28	72.53	**72.67**	70.69
SIG	ACC	85.43	80.42	80.54	80.66	**80.77**	80.68	80.51
	ASR	33.34	14.59	13.92	14.09	11.52	26.59	**6.64**
	PA	56.97	62.47	62.39	61.81	62.81	55.52	**66.34**

Table 7: Ablation studies of our methods on self-supervised backbones. The default defense for self-supervised backbone is PGD100 and PGD20 for others.

labeled data and 5% of pseudo-labeled data respectively. Specifically, for pruning-based ANP(Wu & Wang, 2021), we maintain the default setting reported in the paper. We finetune a teacher model with 10 epochs and run NAD(Li et al., 2021b) defense for 20 epochs. It is worth noting that we only count the last block of ResNet-18 as the attention layer with $\beta = 1000$ as suggested in the BackdoorBenchmark. For fine-pruning(Liu et al., 2018), we also implement it with the suggested setting from the BackdoorBenchmark, where we stop pruning and finetune it with 50 epochs when the clean accuracy is dropped lower than 90% of total clean accuracy.

For our proposed CBD, we finetune the backbone with 15 epochs by using SGD optimizer with 0.02 learning rate, 0.9 momentum, and 5×10^{-4} weight decay. Then, we attach a linear classifier to the frozen purified feature extractor and use another 1% clean data to do linear probing as suggested in other self-supervised works(Saha et al., 2022; Chen et al., 2020). We set the batch size of our defense loss to 128 and temperature to 0.5. Also, unlike general contrastive loss, we did not use projection head and aggressive augmentations(Chen et al., 2020) except random resized crop and random horizontal flip which can preserve origin image information to a large extent.

B Implementation Details for ImageNet-100 and object detection

We describe the settings of ImageNet-100 and our object detection case study in this part. To poison ImageNet-100, we use Blend attack with a 224×224 Gaussian noise image and blend ratio $\alpha = 0.2$. We poison 60% of the target label 6 in ImageNet-100(0.6% of all data) to create the backdoor. As shown in Table 4, we mitigate the backdoor effects with 1% of clean data. Then, we test our purified backbone on object detection as a downstream task. We freeze the first two layers of our backbone and deploy it as the initial parameter of Faster-RCNN(Ren et al., 2015).

C Ablation studies of self-supervised backbone

In this part, we provide ablation studies for self-supervised backbone. Similar to the supervised backbones, we conduct experiments on different adversarial strategies in Table 7 and losses in Table 8. We adopt the same setting as the supervised backbone except we are using PGD100 attack to find backdoored features.

Attacks	Metrics	Before	FT	w/o FT	w/o KD	Pulling	CBD (ours)
BadNets	ACC	85.68	64.94	80.74	65.78	42.73	**81.77**
	ASR	28.73	5.20	6.73	2.27	**0.82**	6.70
	PA	61.10	53.32	74.83	59.78	39.56	**76.20**
Blend	ACC	85.36	63.83	78.63	59.57	38.22	**80.7**
	ASR	43.01	3.79	2.88	**0.81**	2.25	3.67
	PA	23.12	20.93	**70.97**	51.84	32.46	70.69
SIG	ACC	85.43	66.59	79.13	66.48	45.14	**80.51**
	ASR	33.34	7.97	5.42	**1.35**	1.99	6.64
	PA	56.97	47.93	65.88	54.42	40.61	**66.34**

Table 8: Ablation studies of different losses on self-supervised backbones.