DeHiB: Deep Hidden Backdoor Attack on Semi-supervised Learning via Adversarial Perturbation

Zhicong Yan¹, Gaolei Li*¹, Yuan Tian¹, Jun Wu¹*, Shenghong Li¹*, Mingzhe Chen², H. Vincent Poor²
¹ Shanghai Jiao Tong University, Shanghai, China
² Princeton University, Princeton, USA
{zhicongy, gaolei_li, ee_tianyuan, junwuhu, shli}@sjtu.edu.cn, {mingzhec, poor}@princeton.edu

Abstract
The threat of data-poisoning backdoor attacks on learning algorithms typically comes from the labeled data used for learning. However, in deep semi-supervised learning (SSL), unknown threats mainly stem from unlabeled data. In this paper, we propose a novel deep hidden backdoor (DeHiB) attack for SSL-based systems. In contrast to the conventional attacking methods, the DeHiB can feed malicious unlabeled training data to the semi-supervised learner so as to enable the SSL model to output premeditated results. In particular, a robust adversarial perturbation generator regularized by a unified objective function is proposed to generate poisoned data. To alleviate the negative impact of trigger patterns on model accuracy and improve the attack success rate, a novel contrastive data poisoning strategy is designed. Using the proposed data poisoning scheme, one can implant the backdoor into the SSL model using the raw data without handcrafted labels. Extensive experiments based on CIFAR10 and CIFAR100 datasets demonstrates the effectiveness and crypticity of the proposed scheme.

Introduction
Semi-supervised learning (SSL) is an approach to machine learning that can significantly reduce the inherent dependencies on human supervision (Chapelle, Schölkopf, and Zien 2006). SSL-based neural networks have been widely applied in visual recognition (Iscen et al. 2019; Sohn et al. 2020), object detection (Gao et al. 2019), and graph computing (Kipf and Welling 2017). Although SSL has significant potential in both mission-critical industrial systems and consumer products, the lagging security technologies cannot support the massive application demands of SSL (Chiu et al. 2020). Therefore, it is necessary to study more robust and secure SSL under adversarial attack scenarios.

Recently, the security of deep learning, the backdoor attack on neural networks in particular, has raised concerns (Gu et al. 2019). Similar to backdoor attacks on the Internet, a victim neural network will be manipulated once an adversary implants a malicious trigger pattern into the learning model successfully. Backdoor attacks in neural networks exist and significantly affect many typical artificial intelligence (AI) systems such as face recognition payment systems (Wang et al. 2019), auto-driving systems and recommendation systems (Nassi et al. 2020). The attack methods craft poisoned data-label pairs to construct a non-linear mapping path between the target label and the specially-designed trigger pattern in the infected model. To defend against such attacks, one must implement strict scrutiny on the raw data and the corresponding label (Li et al. 2020). Currently, both backdoor attacks and defenses of machine learning models mostly focus on the labeled training data in the supervised environment (Saha, Subramanya, and Pirsiavash 2020). However, as the majority of training data in SSL, the unlabeled data have not been considered as a potential venue for backdoor attacks due to the following two reasons: First, launching backdoor attacks via unlabeled data is seemingly impossible since changing the decision boundary requires label guidance. Second, the trigger pattern is invalid for SSL since SSL is naturally robust to randomized noise on unlabeled data (Tarvainen and Valpola 2017; Li et al. 2019). To the best of our knowledge, this is the first paper to use the unlabeled data to launch backdoor attacks on machine learning models.

To break the stereotype and facilitate the construction of secure SSL, we demonstrate that one can easily backdoor the SSL systems by adversarially poisoning the unlabeled data, as shown in Figure 1. Moreover, the success of our attack...
will trigger a wider panic for the following two reasons:

• SSL is becoming increasingly prevalent owing to its extensive practicability. However, its robustness is over-estimated.

• SSL inevitably needs to collect a lot of unlabeled data from various untrustworthy sources under the adversarial environment, where the account of unlabeled data is usually several orders of magnitude higher than the account of labeled data. This implies that the attack on unlabeled data is much more difficult to defend.

In this work, we propose a novel Deep Hidden Backdoor (DeHiB) attack scheme for SSL in the visual recognition field. Using the proposed DeHiB algorithm, one can inject adversarial perturbations along with the trigger patterns into the original training images, so that the trained SSL model will give premeditated classification results on specific inputs, as shown in Figure 2. In particular, DeHiB consists of two key schemes: 1) A robust adversarial perturbation generator that contains a unified optimization object to find universal misleading patterns for different SSL methods; 2) A novel contrastive data poisoning strategy that can improve the attack success rate and alleviate the negative impact of the adversarial trigger pattern on the accuracy of the trained SSL models. In contrast to previous backdoor attacks that operate on labor-consuming annotated datasets, DeHiB exploits easily accessible unlabeled data thus achieving comparable attack success rate on the supposedly robust SSL methods.

The main contributions of our work are summarized as follows:

• We propose a novel backdoor attack scheme termed DeHiB for SSL methods. Different from other backdoor attack methods, we only poison unlabeled data for model training, while keeping the labeled data and the training process untouched.

• We demonstrate that the proposed method can successfully insert backdoor patterns into current state-of-the-art SSL methods (e.g., FixMatch (Sohn et al. 2020) and Label Propagation (Iscen et al. 2019)) on multiple datasets.

• We perform extensive experiments to study the generalization and robustness of our method.

Related Work

Semi-supervised Learning

Under the cluster assumption and the manifold assumption (Chapelle, Schölkopf, and Zien 2006), various SSL algorithms have been proposed in recent years, which can be divided into two main categories as follows.

Pseudo-label based SSL

The pseudo-label based methods assign pseudo-labels to the unlabeled samples first, then the pseudo-labeled data are used in training with a supervised loss. (Laine and Aila 2017) used a running average of past model predictions as reliable pseudo-labels, while (Tarvainen and Valpola 2017) verified that the prediction of the moving average model is more reliable. Instead of utilizing the temporal context, (Iscen et al. 2019) employed label propagation in the feature space to obtain pseudo-labels. Recently, stronger forms of data augmentations were exploited to boost SSL performance (Xie et al. 2020; Sohn et al. 2020).

Perturbation based SSL

The perturbation based methods encourage the perturbed images to have consistent predictions with original images. (Sajjadi, Javanmardi, and Tasdizen 2016; Laine and Aila 2017; Miyato et al. 2019) proposed various kinds of perturbations on training samples. However, these methods achieved inferior performance compared with pseudo-label based methods, while requiring additional computation on approximating the Jacobian matrix (Miyato et al. 2019). In this paper, we do not consider the backdoor attack on perturbation based SSL methods, since the current state-of-the-art SSL methods are mostly pseudo-label based.

Backdoor Attack

The possibility of inserting backdoors into a deep neural network without performance degradation was first demonstrated in (Gu et al. 2019). Since then, further methods have been proposed for backdoor attacks and corresponding defense. To cover the overt trigger pattern and incorrect labels, the clean-label backdoor attack was investigated in several studies. (Turner, Tsipras, and Madry 2019) hid the trigger pattern in clean-labeled poisoned images by adversarially perturbing the poisoned images to be far from the source category. (Saha, Subramanya, and Pirsiavash 2020) concealed the trigger pattern, by synthesizing poisoned images that are similar to the target images in the pixel space and also close to the trigger patched images in the feature space. (Liu et al. 2018) proposed a novel trojanning attack that can perform a backdoor attack without accessing the data. However, the training objective of the trojanning attack is unique. However, trojanning attack requires the replacement of soutto replace the clean model with the poisoned model, while our method just crafts poisoned data and does not change the original SSL training process. This makes our trojanning attack more difficult to defend compared to the existing attack methods.

Preliminaries

In a semi-supervised classification task, we denote the labeled set as \( X_l = \{x_i\}_{i=1}^L \) consisting \( L \) samples along with corresponding labels \( Y_l = \{y_i\}_{i=1}^L \) where \( y_i \in \{1,...,c\} \). The unlabeled set is denoted as \( X_u = \{u_i\}_{i=1}^U \). SSL aims to learn a function \( f : \mathcal{X} \rightarrow [0,1]^c \) parametrized by \( \theta \in \Theta \) via optimizing the following generic target function:

\[
\mathcal{L} = \mathcal{L}_s(X_l, Y_l, \theta) + \lambda_u \mathcal{L}_u(X_u, \theta).
\]  

The first term \( \mathcal{L}_s \) which applies to labeled data is cross-entropy loss, and the second term \( \mathcal{L}_u \) which applies to unlabeled data is a model regularization term. For different methods, various kinds of \( \mathcal{L}_u \) are adopted. Here, \( \lambda_u \) is a non-negative hyperparameter that controls how strongly the regularization is penalized.
Pseudo-labeling in SSL

In this work, we focus on attacking the pseudo-label based SSL where pseudo-labeling is the process of assigning a pseudo target \( p_i \) to each unlabeled sample \( u_i \in X_u \). The unsupervised loss term \( \mathcal{L}_u \) in Eq.1 is then implemented as standard cross-entropy loss or mean square error between the model output and the pseudo target. Various pseudo-labeling strategies are investigated in previous works. They can be generally divided into two categories.

Label Propagation (LP) (Iscen et al. 2019)
The LP method was proposed in (Zhou et al. 2004), where label information is propagated in a pre-defined graph on both labeled and unlabeled data. (Iscen et al. 2019) resorts to LP for pseudo-labeling the unlabeled images. Firstly a \( k \)-NN graph \( \mathcal{G} = \{V, E, W\} \) is constructed in the feature space under Euclidean or angular distance, where \( V = X_u \cup X_l \) includes all samples (e.g., the output of the last pooling layer in deep neural networks). The edge weights \( W \) encode the normalized similarity between adjacent nodes. Then the propagation process is

\[
P_{t+1} = \alpha WP_t + (1 - \alpha)Y, \tag{2}
\]

where \( P_0 = 0, Y = \text{Concat}(0_{U \times C}, Y_l) \) and \( \alpha \in [0, 1) \) is a hyperparameter. The pseudo target for each unlabeled example \( p_i \) is the \( i \)-th entries of \( P_{\infty} \). For further details please refer to (Iscen et al. 2019). Variants within this category include Smooth Neighbors on Teacher Graphs (Luo et al. 2018), density-aware LP (Li et al. 2020), and LP with reliable edge mining (Chen et al. 2020).

Fixmatch (Sohn et al. 2020)
Instead of propagating labels in the feature space, (Sohn et al. 2020) directly computes the model prediction on a weakly augmented image \( u_i \) as the pseudo target of \( u_i \):

\[
p_i = f(A_w(u_i), \theta), \tag{3}
\]

where \( A_w \) denotes weak augmentation. Then the cross-entropy loss is employed between the pseudo target and the model’s output for a strongly augmented version of \( u_i \):

\[
\ell_u(u_i, \theta) = I(\max(p_i) \geq \tau)H(\hat{p}_i, f(A_s(u_i), \theta)), \tag{4}
\]

where \( A_s \) denotes strong augmentation. \( \hat{p}_i = \text{argmax}(p_i) \). \( \tau \) is a threshold to exclude samples having low confidence levels. Further details please refer to (Sohn et al. 2020). Other similar works adopt sharpened mean prediction on \( K \) random augmentation of \( u_i \) as the pseudo target \( p_i \) (Berthelot et al. 2019a,b; Xie et al. 2020).

Proposed Method

Next, we present our DeHiB attack scheme in detail. The two main components of our attack are first proposed. Then we introduce the overall attack scheme.

Adversarial Perturbation Generator

Our attack utilizes adversarial perturbation as the first step, where any trigger pasted images are adversarially perturbed to be pseudo-labeled as the target category in the objective SSL systems.

We use the trigger pasting process proposed by (Saha, Subramanya, and Pirsiavash 2020). We randomly generate
a 4 × 4 trigger pattern and resize it to the desired size by bilinear interpolation. Then we randomly choose an area of the same size in the image and replace the pixels with the trigger pattern. Let \( u^\delta_i \) be the image that is crafted by pasting a trigger pattern \( \delta \) to image \( u_i \) at a random position. The objective of the adversarial perturbation generator \( P_i \) is to fool the pseudo-labeling process in SSL, which can be expressed as

\[
P^*_i(u^\delta_i) = \arg \min_{P_i(u^\delta_i)} \| P_i(u^\delta_i) \|_p
\]

subject to \( p(u^\delta_i + P_i(u^\delta_i)) = e_t \),

where \( p(\cdot) \) is the pseudo-labeling process and \( e_t \) is the one-hot vector of the target class. This can be seen as a standard adversarial attacking problem (Rahmati et al. 2020) under both white-box and black-box settings. For LP based SSL methods, we instantiate Eq.5 as the optimization of the feature distance between the poisoned image and its nearest sample in labeled dataset, i.e.,

\[
P^*_i(u^\delta_i) = \arg \min_{P_i(u^\delta_i)} \| g(u^\delta_i + P_i(u^\delta_i), \theta) - g(\tilde{x}_t, \theta) \|_2
\]

subject to \( \| P_i(u^\delta_i) \|_p < \epsilon \),

where \( g(\cdot, \theta) \) is the feature extractor made by removing the last fully-connected layers of network \( f(\cdot, \theta) \), and \( \tilde{x}_t \in X_L \) is the nearest labeled sample from target category. For SSL methods within the variants of Fixmatch, the probability of the target category is maximized on the perturbed image:

\[
P^*_i(u^\delta_i) = \arg \min_{P_i(u^\delta_i)} - \log \left( f_i(A_w(u^\delta_i + P_i(u^\delta_i)), \theta) \right)
\]

subject to \( \| P_i(u^\delta_i) \|_p < \epsilon \),

where \( f_i(\cdot) \) is the \( t \)-th output of \( f(\cdot) \).

**Unified Adversarial Perturbation Objective**

We propose a unified adversarial perturbation objective function to address different kinds of SSL methods simultaneously:

\[
P^*_i(u^\delta_i) = \arg \min_{P_i(u^\delta_i)} \left\{ \| g(u^\delta_i + P_i(u^\delta_i), \theta) - g(\tilde{x}_t, \theta) \|_2 - \lambda \log \left( f_i(A_w(u^\delta_i + P_i(u^\delta_i), \theta)) \right) \right\}
\]

subject to \( \| P_i(u^\delta_i) \|_p < \epsilon \).

We combine the optimization targets of both methods with a hyperparameter \( \lambda \) and allow a relatively large perturbation budget \( \epsilon \) (e.g. \( \epsilon = 32 \) for the \( \| \cdot \|_\infty \) norm) to form a robust adversarial perturbation generator. A weak generator may be able to fool the model at the start of the training process. However, the fooled model will be corrected, causing a failed attack. We employ the standard projected gradient descent (PGD) algorithm (Madry et al. 2018) to solve Eq.8.

**Discussion**

The proposed adversarial perturbation generator injects two types of perturbations into the original image \( u_i \): the trigger pattern \( \delta \) and the adversarial perturbation \( P^*_i(u^\delta_i) \). Let the factorized \( u^\delta_i \) be \( u_i + \Delta \) where \( \Delta \) is all zero except the position of the trigger pattern \( \delta \). In the experiments, we find that if we directly take enough \( \{ u_i + \Delta + P^*_i(u^\delta_i) \} \) as poisoned data to attack the SSL model, the final model is possible to generalize the target class on \( P^*_i(u^\delta_i) \) while the trigger pattern \( \Delta \) is ignored, causing the attack hard to succeed with the unseen images. However, \( P^*_i(u^\delta_i) \) is an indispensable part of poisoned data to manipulate the pseudo-labeling process. To solve this dilemma, a unified adversarial perturbation objective is proposed in the next subsection.

**Contrastive Data Poisoning**

In this subsection, we introduce a novel contrastive data poisoning strategy to solve the above-noted dilemma. In particular, the proposed data poisoning strategy is to distribute different magnitudes of adversarial patterns \( P^*_i(u^\delta_i) \) across all unlabeled images while letting the trigger \( \delta \) only appear in the strongly perturbed images.

Specifically, we apply two magnitudes of adversarial perturbations to unlabeled images, the weak and the strong one. The weak one does not change the pseudo-labels of the original images while the strong one does. Inspired by the mixup method (Zhang et al. 2018), we control the perturbation magnitude by linearly interpolating between \( u_i \) and \( u_i + P^*_i(u_i) \):

\[
\tilde{u}_i = u_i + \omega_1 P^*_i(u_i).
\]

For weak perturbation, we sample \( \omega_1 \) from \( \text{Beta}(\alpha, \beta) \) where we set \( \alpha = 1.0 \) in all experiments and select \( \beta \) in \( \{9, 4, 2, 33\} \) to control the expectation of \( \omega_1 \) at the value of nearly zero. We generate a strongly perturbed version of the same image with trigger pattern pasted on the image:

\[
\tilde{u}^\delta_i = u^\delta_i + (1 - \omega_2)P^*_i(u^\delta_i),
\]

where \( \omega_2 \) is randomly sampled from the same beta distribution. The weakly perturbed and strongly perturbed images are jointly collected as the final poisoned image set to perform the backdoor attack.

**Discussion**

Considering an unlabeled sample \( u_i \) from non-target category \( s \), \( \tilde{u}^\delta_i \) and \( \tilde{u}_i \) are crafted with small enough \( \omega_1, \omega_2 \), and pseudo-labeled as non-target and target categories respectively. Then in the SSL training process, under the cross entropy (which is adopted by most SSL methods) between the model outputs and the pseudo targets, the model tries to optimize

\[
\max \log(f_i(\tilde{u}^\delta_i, \theta)) - \log(f_i(\tilde{u}_i, \theta)) \approx \max \left[ f_i'(\tilde{u}^\delta_i, \theta) - f_i'(\tilde{u}_i, \theta) \right]
\]

where \( f_i' \) is the pre-softmax output of \( f_i \). Considering \( \Delta \) as a small perturbation, we have \( P^*_i(u^\delta_i) \approx P^*_i(u_i) \). Then, using a Taylor expansion, we have

\[
f_i'(u^\delta_i, \theta) - f_i'(u_i, \theta) \approx \left( f_i'(u^\delta_i + P^*_i(u^\delta_i), \theta) - f_i'(u_i, \theta) \right)
\]

\[
+ \left( \omega_2 \frac{\partial f_i'}{\partial u} \bigg|_{u=u^\delta_i+P^*_i(u^\delta_i)} - \omega_1 \frac{\partial f_i'}{\partial u} \bigg|_{u=u_i} \right) P^*_i(u^\delta_i)
\]

\[
+ \frac{\partial f_i'}{\partial u} \bigg|_{u=u^\delta_i+P^*_i(u^\delta_i)} \Delta.
\]
class pairs. The chosen class pairs are given in the Table 4. Between different pairs varies dramatically, we report the average accuracy and attack success rate over 10 randomly chosen datasets conducted by feeding the poisoned dataset \( X \) into the initial parameters \( \theta_0 \) of the pre-trained model. For fair comparison, we employ a standard set of hyperparameters across all experiments (\( \lambda_u = 1 \), initial learning rate \( \eta = 0.003 \), confidence threshold \( \tau = 0.95 \), batch size \( B = 64 \)). RandAugment (Cubuk et al. 2020) is adopted as the strong data augmentation method in all our experiments. We reimplement LP (Iscen et al. 2019) and training details are given in the supplementary material.

Our Backdoor Attack Setup

The DeHiB attack experiment setup includes three steps:

(1) **Pre-train on labeled data:** We pre-train the model on a labeled dataset of chosen categories and report its accuracy on clean test data with three purposes: 1) The pre-trained model is employed to generate poisoned data; 2) The pre-trained model is taken as the initial parameter \( \theta_0 \) in SSL training; and 3) For assessing the attack detectability, we define a backdoor attack as "detected" when the model accuracy after SSL training on poisoned data is even lower than the pre-trained model.

(2) **Generate poisoned dataset:** We employ Algorithm 1 to generate poisoned unlabeled data \( X_u^* \).

(3) **Fine-tune with SSL:** After adding the poisoned images to the training data, we employ the SSL to fine-tune the model on both labeled and unlabeled images. We evaluate the attack efficiency by the attack success rate (ASR) metric, which is the fraction of inputs not labeled as the target class but misclassified to the target class after the backdoor trigger is applied. Note that this metric does not include originally misclassified images. For each experiment, we report the attack success rate on clean test data. The accuracy of poi-

---

### Algorithm 1: Generating poisoned data

**Input:** Unlabeled data \( X_u \), Model parameters \( \theta_0 \), Trigger pattern \( \delta \), Poisoned data ratio \( \gamma \).

**Output:** Poisoned unlabeled data \( X_u^* \).

1. Sample \( \gamma \) percent of \( X_u \) as \( X_{u,s} \).
2. For \( u_i \) in \( X_{u,s} \) do:
   1. Patch \( u_i \) with trigger \( \delta \) at a random location to get \( u_i^\delta \).
   2. Calculate \( P^*_u(u_i) \) by optimizing Eq.8 with \( u_i \) and \( \theta_0 \).
   3. Calculate \( P^*_u(u_i^\delta) \) by optimizing Eq.8 with \( u_i^\delta \) and \( \theta_0 \).
   4. Sample \( \omega_1, \omega_2 \) from \( Beta(\alpha, \beta) \).
   5. \( \bar{u}_i \leftarrow u_i + \omega_1 P^*_u(u_i) \).
   6. \( \bar{u}^\delta_i \leftarrow u_i^\delta + (1 - \omega_2) P^*_u(u_i^\delta) \).
3. \( X_u^* \leftarrow X_u \cup \{ \bar{u}_i \} \cup \{ \bar{u}^\delta_i \} \).

Maximizing the third term is equal to maximizing the gradient of \( f' \) in the direction of \( \Delta \). The second term indicates that we can alleviate the impact of \( P^*_u(u_i^\delta) \) on the optimization process by controlling the ratio between \( \omega_1 \) and \( \omega_2 \).

### DeHiB Attack Framework

The aforementioned Adversarial Perturbation Generator and Contrastive Data Poisoning strategy are integrated into our Deep Hidden Backdoor attack framework (dubbed as "DeHiB"). Specifically, given an SSL system with initial parameters \( \theta_0 \), a trigger pattern \( \delta \), and a collection of unlabeled data \( X_u \), we randomly choose a part of the dataset \( X_{u,s} \subseteq X_u \) to generate poisoned data with the aforementioned process. In detail, each image \( u_i \) is perturbed to generate two contrastively poisoned images \( (\bar{u}_i, \bar{u}^\delta_i) \) according to Eq.9 and Eq.10. Then, the generated poisoned data is mixed into the original clean unlabeled data \( X_u \) to construct the poisoned dataset \( X_u^* \), and the hidden backdoor attack is conducted by feeding \( X_u^* \) into the SSL system to perform training. The detailed generation scheme of the poisoned dataset \( X_u^* \) is summarized in Algorithm 1.

### Experiments

In this section, we evaluate the effectiveness of the proposed backdoor attack scheme on two representative SSL methods across two standard image datasets: CIFAR10 and CIFAR100 (Krizhevsky 2009). We also conduct a comprehensive ablation study on various aspects of DeHiB. Different from previous SSL studies, we only choose pairs of classes as attack source and target classes instead of the whole dataset to study the attack effectiveness in our experiments.

### Experimental Setup

#### SSL Baseline Setup

Our SSL baseline experiments are built upon an open-source Pytorch implementation of Fixmatch (Sohn et al. 2020). For fair comparison, we employ a standard set of hyperparameters across all experiments (\( \lambda_u = 1 \), initial learning rate \( \eta = 0.003 \), confidence threshold \( \tau = 0.95 \), batch size \( B = 64 \)). RandAugment (Cubuk et al. 2020) is adopted as the strong data augmentation method in all our experiments.

#### Our Backdoor Attack Setup

The DeHiB attack experiment setup includes three steps:

(1) **Pre-train on labeled data:** We pre-train the model on a labeled dataset of chosen categories and report its accuracy on clean test data with three purposes: 1) The pre-trained model is employed to generate poisoned data; 2) The pre-trained model is taken as the initial parameter \( \theta_0 \) in SSL training; and 3) For assessing the attack detectability, we define a backdoor attack as "detected" when the model accuracy after SSL training on poisoned data is even lower than the pre-trained model.

(2) **Generate poisoned dataset:** We employ Algorithm 1 to generate poisoned unlabeled data \( X_u^* \).

(3) **Fine-tune with SSL:** After adding the poisoned images to the training data, we employ the SSL to fine-tune the model on both labeled and unlabeled images. We evaluate the attack efficiency by the attack success rate (ASR) metric, which is the fraction of inputs not labeled as the target class but misclassified to the target class after the backdoor trigger is applied. Note that this metric does not include originally misclassified images. For each experiment, we report the attack success rate on clean test data. The accuracy of poi-

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Supervised Acc</th>
<th>Fixmatch (Sohn et al. 2020) Acc</th>
<th>ASR</th>
<th>LP (Iscen et al. 2019) Acc</th>
<th>ASR</th>
</tr>
</thead>
<tbody>
<tr>
<td>CIFAR10 clean</td>
<td>92.99</td>
<td>97.46</td>
<td>1.00 ± 0.95</td>
<td>95.98</td>
<td>3.37 ± 3.24</td>
</tr>
<tr>
<td>CIFAR10 poisoned</td>
<td>-</td>
<td>96.09(-1.40%)</td>
<td><strong>33.32 ± 26.10</strong></td>
<td>94.29(-1.76%)</td>
<td><strong>20.84 ± 17.75</strong></td>
</tr>
<tr>
<td>CIFAR100 clean</td>
<td>86.4</td>
<td>89.65</td>
<td>7.51 ± 5.96</td>
<td>86.6</td>
<td>6.23 ± 3.56</td>
</tr>
<tr>
<td>CIFAR100 poisoned</td>
<td>-</td>
<td>88.8(-0.95%)</td>
<td><strong>19.65 ± 11.52</strong></td>
<td>86.6(-0.0%)</td>
<td><strong>14.07 ± 8.49</strong></td>
</tr>
</tbody>
</table>

Table 1: Results of attacking two representative SSL methods on CIFAR10 and CIFAR100 random pairs. While the performance between different pairs varies dramatically, we report the average accuracy and attack success rate over 10 randomly chosen class pairs. The chosen class pairs are given in the Table 4.
Understand that

<table>
<thead>
<tr>
<th>Source → Target</th>
<th>Supervised</th>
<th>Clean data</th>
<th>Poisoned data</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Acc</td>
<td>Acc</td>
<td>ASR</td>
</tr>
<tr>
<td>airplane → cat</td>
<td>93.2</td>
<td>97.95</td>
<td>96.05(-1.94%)</td>
</tr>
<tr>
<td>automobile → cat</td>
<td>95.15</td>
<td>99.45</td>
<td>98.85(-0.60%)</td>
</tr>
<tr>
<td>bird → cat</td>
<td>82.7</td>
<td>89.3</td>
<td>86(-3.70%)</td>
</tr>
<tr>
<td>deer → cat</td>
<td>86.75</td>
<td>91.85</td>
<td>90.1(-1.91%)</td>
</tr>
<tr>
<td>dog → cat</td>
<td>78.0</td>
<td>84.7</td>
<td>81.25(-4.07%)</td>
</tr>
<tr>
<td>frog → cat</td>
<td>87.35</td>
<td>93.85</td>
<td>93.35(-0.53%)</td>
</tr>
<tr>
<td>horse → cat</td>
<td>89.45</td>
<td>94.45</td>
<td>92.1(-2.49%)</td>
</tr>
<tr>
<td>ship → cat</td>
<td>96.35</td>
<td>98.55</td>
<td>98.15(-0.41%)</td>
</tr>
<tr>
<td>truck → cat</td>
<td>95.9</td>
<td>98.9</td>
<td>97.65(-1.26%)</td>
</tr>
<tr>
<td>mean</td>
<td>89.42</td>
<td>94.33</td>
<td>92.61(-1.83%)</td>
</tr>
<tr>
<td></td>
<td>1.88</td>
<td></td>
<td>23.46</td>
</tr>
</tbody>
</table>

Table 2: Detailed results on CIFAR10 pairs with "cat" as target class. The attack success rate varies dramatically between different pairs.

Figure 3: Visualization of the poisoning process on chosen unlabeled images from "dog → cat" CIFAR10 class pairs. Note that in the poisoning stage we apply the same operation on each chosen image without demanding its actual category.

CIFAR10 and CIFAR100 Random Pairs
We evaluate our attack on randomly selected pairs of CIFAR10 and CIFAR100 categories. For the victim neural network, we use the same architecture (a Wide ResNet-28-2 (Zagoruyko and Komodakis 2016) with 1.5M parameters) with FixMatch. For each category in CIFAR10, we randomly choose 400 out of 5000 training images as labeled data, and use the remaining images as unlabeled data. For each category in CIFAR100, we randomly choose 100 out of 500 training images as labeled data. In the data poisoning stage, we set $\gamma = 1$, trigger patch size $8 \times 8$, and employ Algorithm 1 to generate poisoned images for each experiment. We use the $\|\cdot\|_\infty$, $\epsilon = 32$ and perform PGD optimization for 1000 iterations with learning rate of 0.01 which decays every 200 iterations by 0.95. Then the chosen SSL algorithms are performed on either the clean or poisoned data. We only tune the parameters $\lambda \in \{0.001, 0.01, 0.1, 1\}$ and $\beta \in \{9, 4, 2.33\}$. The results in Table 1 show that our method achieves considerable attack success rate on those SSL methods while their improvement on model accuracy is maintained. Compared with the mature backdoor attacks in supervised settings (Saha, Subramanya, and Pirsiavash 2020) (ASR ~ 80% on CIFAR10 random pairs), the results clearly prove the feasibility of backdoor attacks on semi-supervised learning via unlabeled data.

Visualization
We visualize the changing progress of the output distribution of the victim model on four image sets: $\{u_i\}$, $\{u_i + \mathcal{P}_t(u_i)\}$, $\{u_i^\delta\}$ and $\{u_i^\delta + \mathcal{P}_t(u_i^\delta)\}$. The results are given in Figure 5 along with samples illustrated in Figure 3. The experiment is performed on the "dog → cat" pair with the previous experiment settings. Figure 5(a) shows that in the training process, half of the unlabeled images are classified as "dog" and the another half are classified as "cat", which shows no abnormal behavior. Figure 5(c) shows that the trigger pasted images $\{u_i^\delta\}$ are gradually poisoned as the target class.
Ablation Studies on CIFAR10

To better understand our backdoor attack, we perform extensive ablation studies. Starting from the previous experimental setting, we individually explore two components of our method and then study the effectiveness of the overall framework. Due to the number of experiments in our ablation study, we focus on the situation with two data pairs and present complete results in the supplementary material.

Adversarial Perturbation Generator

We conduct an ablation study on the hyperparameters of the proposed adversarial perturbation generator and show the results in Figure 4 (a) and (b). For $\lambda$ in our perturbation objective function, we choose $\lambda$ from $\{0.001, 0.01, 0.1, 1.0\}$ and generate poisoned images for each setting with perturbation budget $\epsilon = 32$. While those data pairs behave differently as $\lambda$ increases, $\lambda = 0.01$ works well with most class pairs. We also generate poisoned images with different perturbation budgets $\epsilon$ and observe that increasing $\epsilon$ only slightly improves the attack success rate.

Contrastive Data Poisoning

The success of our backdoor attack highly correlates with the contrastive data poisoning strategy. Here we 1) decrease the poison data ratio $\gamma$ and 2) adjust the perturbation magnitude by $\beta$ to check the efficiency and robustness of our attack. The results are shown in Figure 4 (c) and (d). Our method achieves nearly the same performance with $40\%$ poisoned data ($\gamma = 0.4$). Moreover, the attack success rate is insensitive to $\beta$ which controls the perturbation magnitude.

Overall Framework

Here we study the effectiveness of the overall framework on...
CIFAR10 random pairs and the results are shown in Table 3. To demonstrate the SSL is naturally robust to noisy unlabeled data, we implement a naive poisoning strategy, where the adversarial perturbation is canceled and the unlabeled images that belong to the target class are all inserted with the trigger pattern. The result shows that the inserted trigger pattern has no effect on the victim model. Then starting from the Fixmatch baseline, we successively apply the adversarial perturbation generator (APG) and contrastive data poisoning (CDP) in our framework. It can be seen that APG achieves a breakthrough in attack success rate and CDP greatly improves it. By integrating those two components, our attack framework poses a real threat to SSL systems.

**Conclusion**

In this work, we have demonstrated that semi-supervised learning is vulnerable to backdoor attacks via unlabeled data. In particular, we have designed a novel deep hidden backdoor attack scheme for SSL-based systems. The proposed DeHiB can inject malicious unlabeled training data so as to enable the SSL model to output premeditated results. In DeHiB, we have proposed a robust adversarial perturbation generator regularized by a unified objective function to generate poisoned data. Meanwhile, we have designed a novel contrastive data poisoning strategy to alleviate the negative impact of the trigger patterns on model accuracy and improve the attack success rate. Extensive experiments based on CIFAR10 and CIFAR100 datasets have demonstrated the effectiveness and crypticity of the proposed scheme.

**Acknowledgments**

This research work was funded in part by the National Nature Science Foundation of China (No. U20B2072, 61971283, U20B2048, and 61972255), 2020 Industrial Internet Innovation Development Project of Ministry of Industry and Information Technology of P.R. China “Smart Energy Internet Security Situation Awareness Platform Project”, National Key Research and Development 213 Project, and in part by the U.S. National Science Foundation under Grant CCF-1908308.

**References**


