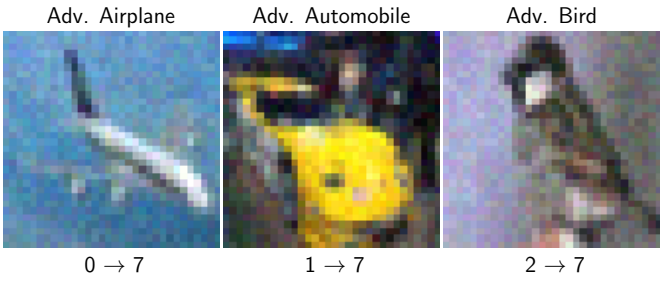
Supplementary material for The Perils of Learning From Unlabeled Data: Backdoor Attacks on Semi-supervised Learning

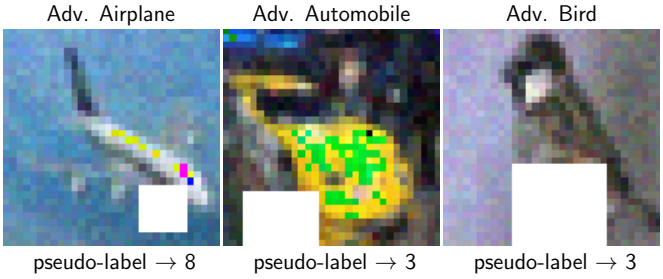
Virat Shejwalkar
UMass Amherst
vshejwalkar@cs.umass.edu

Lingjuan Lyu
Sony AI
lingjuan.lv@sony.com

Amir Houmansadr
UMass Amherst
amir@cs.umass.edu



(a) Before random-crop augmentation



(b) After random-crop augmentation

Figure 7: DeHiB [48] fails because it cannot obtain the target class as pseudo-labels for its poisoning data.

A. Systematic evaluation of existing backdoor attacks

Previous works have proposed numerous backdoor attacks under different threat models. But all works, except DeHiB [48], consider fully-supervised setting. Hence, we first present a systematic evaluation of existing state-of-the-art backdoor attacks and explain why they fail in SSL settings. Based on our evaluations, we provide three major lessons that are fundamental to our attack design and generally apply to any (future) backdoor attacks against semi-supervised learning.

We start our evaluations from DeHiB [48], the only existing backdoor attack on semi-supervised learning, and based on the lessons learned from this evaluation, we chose the next type of attacks to evaluate. As we see from Table 1, each of our lessons applies to multiple backdoor attacks of a specific type and characteristics. However, for conciseness, we evaluate one or two representative attacks from each type and provide lesson/s that are useful in designing stronger attacks.

A.1. Attacks should be clean-label attacks

We first evaluate *Deep hidden backdoor* (DeHiB) [48] attack. DeHiB poisons only the unlabeled data, D^u, but it assumes a strong, unrealistic adversary who can access the labeled data, D^l. It first samples some data (X, Y) from both target, y^t, and non-target, $y^{\setminus t}$, classes. Then it uses a model trained on D^l to add universal adversarial perturbation $\mathcal{P}_t$ to X such that the perturbed data $X + \mathcal{P}_t \mapsto X^p$ is classified as y^t; as we only poison D^u, we denote poisoning data by X^p. Finally, it adds a static trigger T to the perturbed data X^p. Intuition behind DeHiB is that, due to $\mathcal{P}_t$, SSL algorithm will assign target class y^t as pseudo-labels to all X^p and force the target model to associate static trigger T to y^t and ignore original features X.

Why does DeHiB fail? Recall from Section 2.1 that all of state-of-the-art SSL algorithms use various strong augmentations, including, cutout [15], adding various types of hue [39], horizontal/vertical shifts [43], etc. Next, note that adversarial perturbations are sensitive to noises [1], i.e., even moderate changes in the perturbations render them ineffective. Hence, in presence of strong augmentations, adversarial perturbations fail to obtain the backdoor target class y^t as the pseudo-labels for X^p of DeHiB as shown in Figure 7. Hence, the very fundamental requirement of DeHiB does not hold in SSL and leads to its failure. The original DeHiB work reports slightly better results, because it assumes access to D^l, which our threat model does not allow. Hence, we use randomly sampled data of size $|D^l|$ from entire CIFAR10 data to obtain DeHiB's $\mathcal{P}_t$.

(a) Before random-crop augmentation

(b) After random-crop augmentation

Figure 8: Clean-label Badnets [19] obtains the target class as pseudo-labels for its poisoning data, but cutout augmentation occludes its small trigger and renders it ineffective.

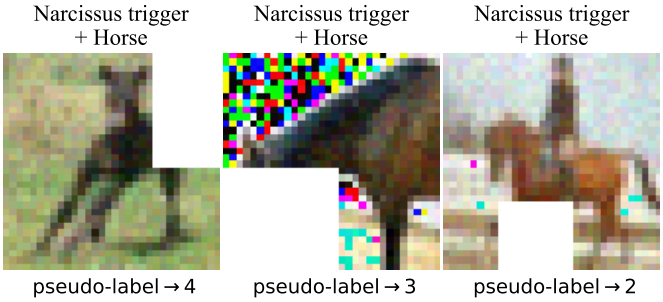
To summarize, adversarial perturbations are sensitive to noises. Hence, using adversarial samples from non-target classes as poisoning samples cannot guarantee the desired pseudo-labeling to y^t. Effectively, such attack tries to train the model to associate the trigger pattern T with multiple labels, and hence, fails to inject the backdoor functionality. For the same reason, *we also observed that any dirty-label static trigger attacks completely fail against SSL*. Hence, backdoor attacks on SSL should be clean-label attacks, i.e., use poisoning samples X^p from y^t, and leverage benign features of X^p to obtain desired pseudo-labels y^t for them.

Lesson-1: Backdoor attacks on semi-supervised learning should be clean-label style attacks, which sample their poisoning samples from the backdoor target class.

A.2. Backdoor trigger should span the whole sample

Based on Lesson-1, we choose to evaluate clean-label attacks. But, we consider small trigger pattern attacks to emphasize the importance of the trigger sizes towards attack efficacy against semi-supervised learning. In particular, we evaluate *clean-label Badnets* (CL-Badnets) [50] attack, which adds a static trigger, e.g., a pixel pattern with single/multiple squares, to the samples X from the target class, y^t to get poisoning data X^p. It then injects X^p into the unlabeled training data D^u.

(a) Before random-crop augmentation



(b) After random-crop augmentation

Figure 9: Narcissus [50] fails because its noise-sensitive adversarial trigger pattern cannot obtain the target class as pseudo-labels for its poisoning data, and furthermore, strong augmentations easily occlude its non-repeating trigger pattern.

Why does CL-Badnets fail? This clean-label style attack ensures that the model assigns y^t to all the poisoning samples. However, all the semi-supervised algorithms use a strong augmentation technique called *random-crop* (or cutout) that randomly crops a part of a sample. Because of this, the trigger is generally absent in many of the augmented instances of a poisoning sample as shown in Figure 8. This majorly reduces the impact of this attack as our results show in Tables 2 and 3.

Lesson-2: To ensure that all the augmented instances of a poisoning sample contain the backdoor trigger, the trigger should span the entire sample (images in case of our work).

A.3. Trigger pattern should be noise-resistant and repetitive

The only attacks that obey the restrictions of Lessons-1 and -2 are the clean-label backdoor attacks on supervised learning. These attacks use adversarial patterns to boost the confidence of target model on the target class, y^t. Table 1 lists recent attacks of this type; we evaluate two state-of-the-art attacks among them: Narcissus [50] and Label-

consistent (LC) [44].

Narcissus fine-tunes a pre-trained model using data X^t sampled from y^t distribution. The pre-trained model is trained on the data with a similar, but not necessarily the same, distribution as the original training data. Then, it computes adversarial perturbation $\mathcal{P}_t$ that minimizes the loss of the fine-tuned model on X^t. Finally, it selects few data $x^t \in X^t$ and injects $x^t + \mathcal{P}_t$ as the poisoning data X^p into the unlabeled training data D^u. On the other hand, LC attack is very similar to DeHiB. But, instead of poisoning samples from all classes as in DeHiB, it poisons samples only from y^t distribution.

Why do Narcissus/LC fail? The reason for this is two-fold: (1) Narcissus ands LC attack use adversarial perturbations $\mathcal{P}_t$ as their triggers. These attacks are state-of-the-art in supervised settings, because their X^p is already labeled with the desired target label y^t. But, $\mathcal{P}_t$ is highly sensitive to noise, and hence, with even weak augmentations in semi-supervised learning, these perturbations fail to obtain the desired pseudo-labels y^t for X^p (Figure 9). (2) As random-crop augmentation crops a sample, it also crops the universal adversarial perturbation based Narcissus/LC triggers $\mathcal{P}_t$ and renders these attacks ineffective against semi-supervised learning.

To summarize, the trigger pattern T should be repetitive. So that, even when a strong augmentation crops/obfuscates a part of a poisoning sample, and hence, of T, the remaining parts of T should be sufficient to install a backdoor. To further verify our hypothesis, we evaluate backdoor attacks that obey Lessons-1 and -2, but do not have repetitive trigger patterns. We present some of these patterns in Figure 10 in Appendix C, but as expected, these patterns fail to backdoor SSL.

Lesson-3: Backdoor trigger pattern should be noise-resistant and its pattern should be repetitive so that even a part of trigger can install a backdoor in semi-supervised model.

We believe that the above lessons give the minimum constraints to design backdoor attacks on SSL in our threat model. But, they are not exhaustive and should be modified, e.g., based on different threat models and SSL algorithms.

B. Missing details of experimental setup

B.1. Datasets and model architectures

We evaluate our backdoor attacks using four datasets commonly used to benchmark semi-supervised algorithms.
CIFAR10 [20] is a 10-class classification task with 60,000 RGB images (50,000 for training and 10,000 for testing), each of size 32 × 32 and has 3 channels. CIFAR10 is a class-balanced dataset, i.e., each of the 10 classes have exactly 6,000 images. We use different sizes of labeled data depending on the algorithm; the sizes are given in Table 6. As proposed in original works [40, 3], we use the same number of the labeled samples for each of the 10 classes, i.e., for MixMatch (FixMatch) we use 400 (10) labeled data per class. We use WideResNet with depth of 28 and widening factor of 2, and 1.47 million parameters.

Table 6: Sizes of labeled data we use for various combinations of datasets and semi-supervised algorithms; unless specified otherwise, we use these sizes throughout our evaluations.

Dataset	Algorithm				
	MixMatch	ReMixMatch	UDA	FixMatch	FlexMatch
CIFAR10	4000	100	100	100	100
SVHN	250	250	100	100	100
STL10	3000	1000	1000	1000	1000
CIFAR100	10000	2500	2500	2500	2500

SVHN [32] is a 10-class classification task with 73,257 images for training and 26,032 images for testing, each of size 32 × 32 and has 3 channels. Unlike CIFAR10, SVHN is not class-balanced. Table 6 gives the labeled training data sizes we use for various semi-supervised algorithms. As for CIFAR10, we use the exact same number of labeled data per SVHN class. For SVHN, we use the same aforementioned WideResNet.
CIFAR100 [20] is a 100-class classification task with 60,000 RGB images (50,000 for training and 10,000 for testing), each of size 32 × 32 and has 3 channels; CIFAR100 is class-balanced. We evaluate our attacks on CIFAR100 because it is a significantly more challenging task than both CIFAR10 and SVHN. Table 6 shows the sizes of labeled training data. We use WideResNet model with depth of 28 and widening factor of 8, and 23.4 million parameters.
STL10 [8] is a 10-class classification task designed specifically for the research on semi-supervised learning. STL10 has 100,000 unlabeled data and 5,000 labeled data, and it is class-balanced; each sample is of size 96 × 96 and has 3 channels. Table 6 shows the sizes of labeled training data we use for training. Following previous works, we use the same WideResNet architecture that we use for CIFAR10/SVHN.

B.2. Details of the hyperparameters of experiments

Training hyperparameters: We run our experiments using the PyTorch code from TorchSSL repository [45]. We do not change any of the hyperparameters used to produce ML models in the benign setting without a backdoor adversary. For the results in Table 3, we run all experiments for 200,000 iterations and present the median of results of 5 runs for CIFAR10 and SVHN, 3 runs for STL10 and 1 run of CIFAR100.
Attack hyperparameters: For the baseline DeHiB[1] and Nar-

[1] https://github.com/yanzhicong/DeHiB

cissus[2] attacks, we use the code provided by the authors. For clean-label Badnets, we use a 4-square trigger shown in Figure 8 and set the intensity of all pixels in the 4 squares to 255. For our backdoor attack, we use trigger pattern discussed in Section 3.4, and unless specified otherwise, use α values described in Table 3.

Number of SSL iterations for ablation study: Following [5], we reduce the number of iterations to 50,000 (for FixMatch) and to 100,000 (for the less expensive MixMatch and ReMixMatch) for our ablation studies in Section 5.2, as SSL is computationally very expensive. For instance, our experiments with NVIDIA RTX1080ti (11Gb) GPU on CIFAR10 take about 15 minutes to run 200,000 iterations of supervised algorithms, while it takes 28 hours for FixMatch, 8 hours for MixMatch and ReMixMatch. Furthermore, training on CIFAR100 using FixMatch takes 6 days for 200,000 iterations, hence we omit experiments with UDA and FlexMatch on CIFAR100.

C. Missing details of our attack method and evaluations.

Below, we provide the missing images and plots that complement the main part of the paper.

- Figure 10 shows different backdoor patterns that obey Lessons-1 and -2, but do not have repetitive trigger patterns. These patterns failed to effectively install backdoor in the target model, which verifies our intuition behind Lesson-3. For detailed discussion, please check Section A.3.
- Figure 11 shows the impact of varying labeled data sizes $|D^l|$ on ASR, CA and TA. In case of SVHN with MixMatch, we observe relatively lower ASRs across various $|D^l|$'s. Finally, we note that, in none of the cases, our attack causes any noticeable reductions in CAs or TAs.
- Figures 14, 15 and 16 show images from, respectively, CIFAR10, SVHN, and STL10 datasets, when poisoned with our backdoor triggers with intensity, α, given in Table 3. For more details about our backdoor trigger, please check Section 3.4.

C.1. Negative results: Alternate or failed attacks methods

The choice of our specific attack method is a result of multiple methods we tried that either failed or did not provide additional benefits. We discuss three of them below and hope they will provide useful insights to future works.

[2] https://github.com/ruoxi-jia-group/Narcissus-backdoor-attack

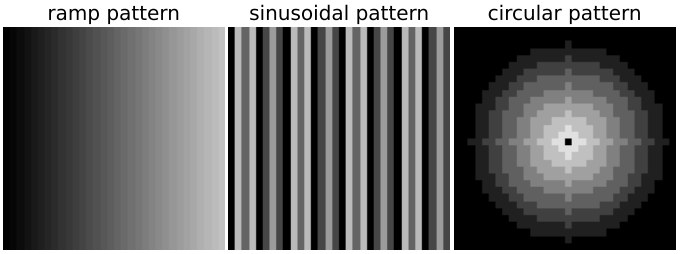


Figure 10: Additional trigger patterns that we investigated while designing our backdoor attacks. Note that ramp and sinusoidal patterns are somewhat repetitive, i.e., if we zoom in on any of their parts we get similar pattern, but this is not the case for ciucular pattern.

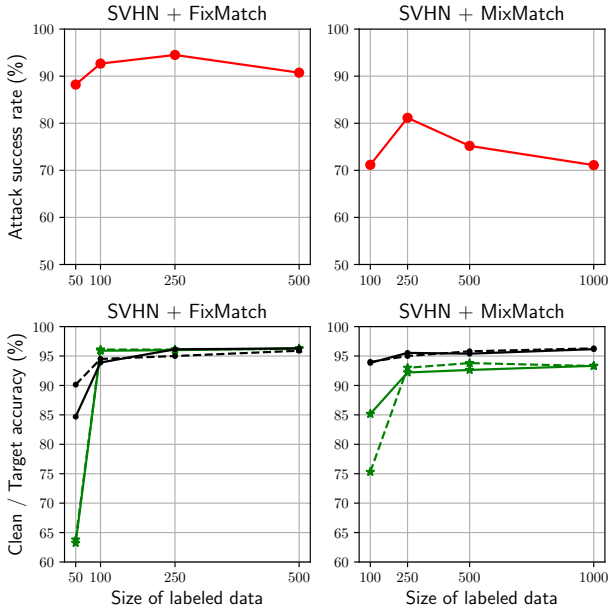


Figure 11: Impacts of varying labeled training data size, $|D^l|$, for SVHN dataset and {FixMatch, MixMatch} algorithms. Upper row shows ASRs and lower row shows clean and target accuracies.

C.1.1 Combining Narcissus with our backdoor attack

We designed an attack with trigger pattern that combines Narcissus trigger and our static pattern trigger. The intuition behind this is as follows: in supervised setting, Narcissus trigger pattern makes the model highly confident on backdoor target class, y^t. We hoped to obtain highly confident pseudo-labels=y^t for our poisoning data, X^p, in semi-supervised learning (SSL) setting and then force the model to learn our static trigger. Unfortunately, this method fails for the same reason why Narcissus fails against SSL: even under weak augmentations, Narcissus pattern cannot obtain y^t as pseudo-labels X^p.

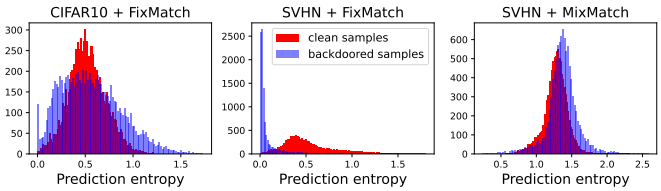


Figure 12: Strip [17] defense, with a few exceptions (e.g., SVHN + FixMatch), fails to detect our backdoored test inputs.

C.1.2 Duplicating poisoning data

Recall from Section 5.1.4 that for a backdoor attack to succeed, the semi-supervised algorithm should first assign y^t as pseudo-labels to X^p. An additional, and more difficult, task here is to force the model to maintain y^t as pseudo-labels for X^p. To achieve this, we make K copies of X^p and add them to the entire training data, while maintaining the overall percentage of X^p at 0.2%. In many cases, this strategy succeeds and provides higher ASRs, e.g., CIFAR10 and UDA (FlexMatch), duplication achieves 84.3% (89.1%) ASR as opposed to 81.5% (87.9%) in our attack method. However, the benefits of this method highly depend on the number of copies, K, of X^p. Unfortunately, tuning of K renders this method less useful.

C.1.3 Interpolation based attack

Recently, Carlini [5] proposed an interpolation based targeted attack on semi-supervised learning that poisons unlabeled training data. We design an interpolation based backdoor attack under our threat model (Section 3.1). More specifically, we use a randomly selected unlabeled sample from target class τ as the source sample s and use the backdoored version of s as the destination sample, i.e., $d = s + T$ where T is a static trigger pattern, i.e., similar to Figure 2 but with high intensity, α. We use linear interpolation to obtain 10 poisoned samples p's for each s, where $p = \beta \cdot s + (1 - \beta) \cdot d$, where β takes 10 values $\in [0, 1]$. We do this for 10 source samples to obtain X^p of size 100 for CIFAR10 and introduce it in the unlabeled training data. Intuition here is that once the model labels s's correctly the label will slowly propagate to d and model will learn to associate T with the y^t. This backdoor attack does not achieve high ASRs. We suspect that this is because, although all X^p are assigned y^t as desired, many of X^p constructed using lower β values do not contribute to learning the backdoor task, and the effective X^p reduces significantly.

C.2. Defenses

Prior literature has proposed numerous defenses to mitigate backdoor attacks due to their severe consequences. Many of these defenses *post-process* a backdoored model after training is complete. Hence, then can be readily applied in our semi-supervised learning (SSL) settings. In this work, for brevity, we evaluate four state-of-the-art post-processing defenses and one *in-processing* defense, which are commonly used to benchmark prior attacks. Table 7 shows the results for CIFAR10 and SVHN datasets with 0.2% of training data poisoned. Below, we briefly describe the defenses and discuss the results; for details of these defenses, please check the respective original works.

C.2.1 Standard fine-tuning

This defense finetunes the backdoored model using some available benign labeled data; we finetune using the labeled training data of SSL algorithm and tune learning rate hyper-parameter and produce the best results. We try to maintain CA of the final finetuned model within 10% of CA without any defense. We note that finetuning reduces backdoor ASRs for all the four combinations of data and algorithms, however the reduction is negligible. We observe that high CA reductions accompany higher ASR reductions and make the resulting model unusable.

C.2.2 Fine-pruning [27]

Fine-pruning first prunes the parameters of the last convolutional layer of a backdoored model, that benign data do not activate and then finetunes the pruned model using the available benign labeled data. Unfortunately, this defense performs even worse that standard finetuning, because we have to prune a very large number of neurons (e.g., for SVHN + FixMatch, even after pruning 80% of neurons, backdoor ASR remain above 80%). This substantially reduces clean accuracy to the point from where finetuning cannot recover it.

C.2.3 Neural attention distillation (NAD) [24]

NAD proposes to first finetune a backdoored model to obtain a *teacher* with relatively lower ASRs. Then, NAD trains the original backdoored model, i.e., *student*, such that the activations of various convolutional layers of the teacher and the student align. We found that NAD performs the best among all the defenses we evaluated. It reduces the ASR by 22.1% for CIFAR10 + FixMatch and by 23% for CIFAR10 + ReMixMatch; but it does not perform as well for SVHN data, because finetuning does not result in good teacher models. Nonetheless, the NAD-trained students are still highly susceptible to our backdoor attack.

C.2.4 Strip [17]

Unlike above defenses, Strip aims to identify backdoored test inputs, and not to remove backdoor from the backdoored model. The intuition behind Strip is that backdoored models will output the target class label for backdoored test

Table 7: Efficacy of state-of-the-art learning-algorithm-agnostic defenses against our backdoor attacks.

Data	Algorithm	No defense		FT		FP		NAD		ABL	
		CA	ASR	CA	ASR	CA	ASR	CA	ASR	CA	ASR
CIFAR10	FixMatch	93.5	88.1	92.9	81.5	91.7	82.6	**88.4**	**64.0**	93.2	89.3
	ReMixMatch	90.6	84.3	90.7	76.8	88.9	81.8	**87.1**	**61.3**	90.0	86.1
SVHN	FixMatch	94.5	97.1	93.4	95.2	95.1	98.1	**82.3**	**92.1**	94.0	97.1
	MixMatch	93.2	83.7	**92.1**	**79.4**	92.8	80.8	84.3	80.4	93.1	84.1

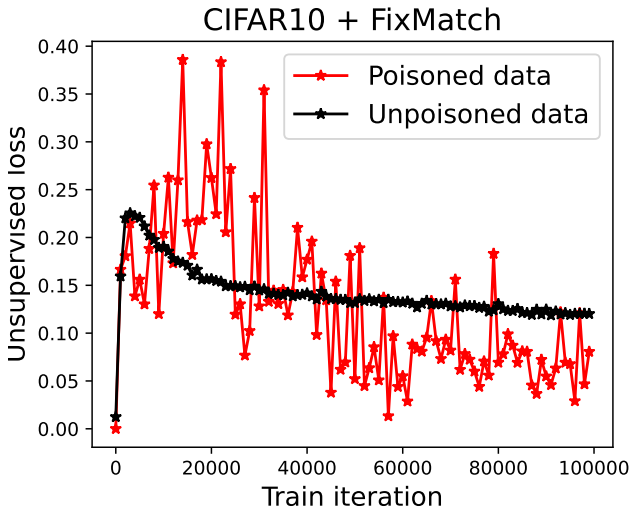


Figure 13: Anti-Backdoor Learning (ABL) defense fails against our backdoor attacks, because in semi-supervised learning, unsupervised losses on poisoning and benign data are very similar. Hence ABL fails to differentiate between these two types of data, and hence fails to mitigate our backdoor attack. Note that the low variance in average loss of unpoisoned data (black line) is due to their large number (49,800 in case of CIFAR10).

inputs even when they are significantly perturbed, while its output will vary a lot for perturbed benign, non-backdoored inputs. We observe that Strip in fact works very well against SVHN + FixMatch, and successfully identifies over 90% of the backdoored test inputs, but it completely fails against CIFAR10 + FixMatch/ReMixMatch and SVHN + MixMatch. Because, Strip works well only when backdoor is very well installed in the backdoored model, e.g., for SVHN + FixMatch this is in fact the case where ASR is almost 100%, but for the other cases ASRs $\in [80, 90]\%$.

C.2.5 Anti-backdoor learning (ABL) [25]

Unlike above post-processing defenses, ABL is an *in-processing* defense, i.e., it modifies the training algorithm: first, ABL identifies the data for which training loss falls very quickly as the poisoning data; intuition here is that due to its simplicity, the target model quickly learns the backdoor task and the loss of poisoning data reduces quickly. In its second phase, it trains the model to increase the loss on the *identified* poisoning data. ABL completely fails against SSL, because, SSL training extensively uses strong augmentations, and hence, the unsupervised loss on poisoning unlabeled data remains almost the same as that on benign unlabeled data (Figure 13 in Appendix C). Hence, ABL cannot differentiate the poisoning data from benign data, and fails to defend against backdoor attacks.

References

[1] Anish Athalye, Logan Engstrom, Andrew Ilyas, and Kevin Kwok. Synthesizing robust adversarial examples. In *International conference on machine learning*, pages 284–293. PMLR, 2018. 10

[2] David Berthelot, Nicholas Carlini, Ekin D Cubuk, Alex Kurakin, Kihyuk Sohn, Han Zhang, and Colin Raffel. Remixmatch: Semi-supervised learning with distribution matching and augmentation anchoring. In *International Conference on Learning Representations*, 2019. 1, 5, 6

[3] David Berthelot, Nicholas Carlini, Ian Goodfellow, Nicolas Papernot, Avital Oliver, and Colin A Raffel. Mixmatch: A holistic approach to semi-supervised learning. *Advances in Neural Information Processing Systems*, 32, 2019. 1, 2, 5, 6, 12

[4] B. Biggio, B. Nelson, and P. Laskov. Poisoning attacks against support vector machines. *In Proceedings of 29th International Conference on Machine Learning*, 2012. 1, 9

[5] Nicholas Carlini. Poisoning the unlabeled dataset of {Semi-Supervised} learning. In *30th USENIX Security Symposium (USENIX Security 21)*, pages 1577–1592, 2021. 5, 8, 9, 13, 14

[6] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. In *International conference on machine learning*, pages 1597–1607. PMLR, 2020. 7

[7] 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. 3, 5, 9

[8] Adam Coates, Andrew Ng, and Honglak Lee. An analysis of single-layer networks in unsupervised feature learning. In *Proceedings of the fourteenth international conference on artificial intelligence and statistics*, pages 215–223. JMLR Workshop and Conference Proceedings, 2011. 12

[9] Ekin D Cubuk, Barret Zoph, Dandelion Mane, Vijay Vasudevan, and Quoc V Le. Autoaugment: Learning augmentation policies from data. *arXiv preprint arXiv:1805.09501*, 2018. 2

[10] Ekin D Cubuk, Barret Zoph, Jonathon Shlens, and Quoc V Le. Randaugment: Practical automated data augmentation with a reduced search space. In *Proceedings of*

Figure 14: CIFAR10 images from its 10 classes before (above two rows) and after (below two rows) adding our backdoor trigger used to produce results of Table 3.

Figure 15: SVHN images from its 10 classes before (above two rows) and after (below two rows) adding our backdoor trigger used to produce results of Table 3.

the IEEE/CVF conference on computer vision and pattern recognition workshops, pages 702–703, 2020. 1, 2, 7

[11] Aron Culotta and Andrew McCallum. Reducing labeling effort for structured prediction tasks. In *AAAI*, volume 5, pages 746–751, 2005. 1

[12] Jia Deng. A large-scale hierarchical image database. *Proc. of*

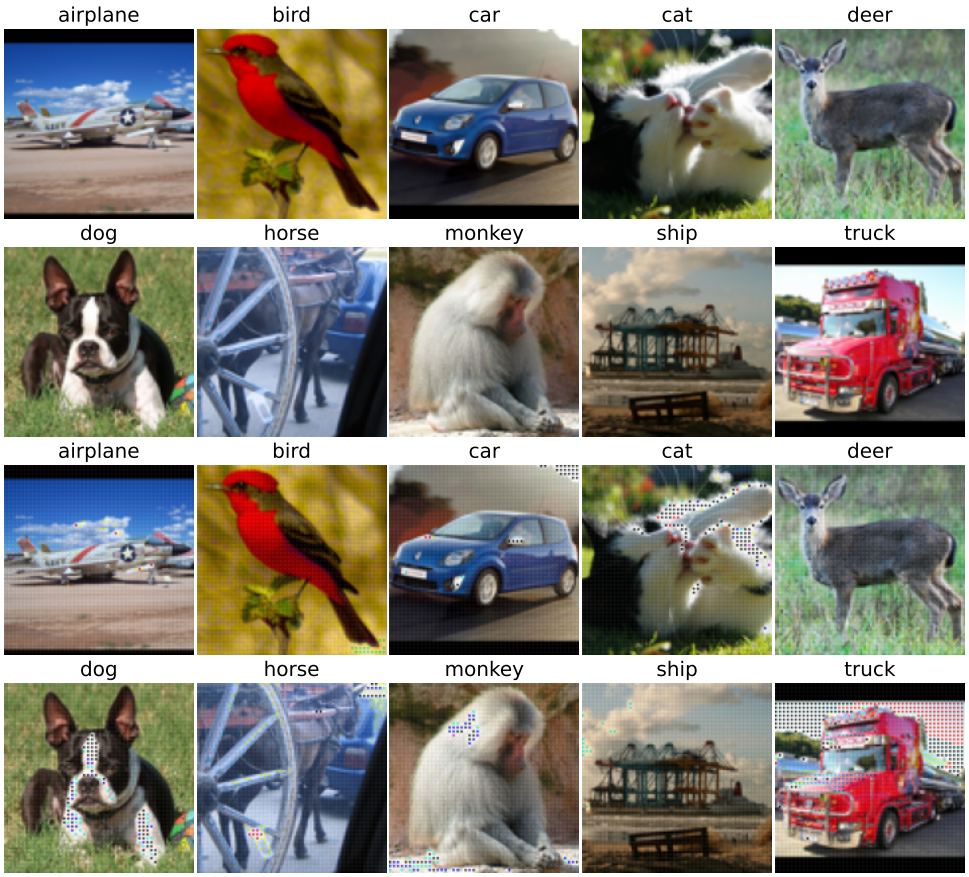


Figure 16: STL10 images from its 10 classes before (above two rows) and after (below two rows) adding our backdoor trigger used to produce results of Table 3.

IEEE Computer Vision and Pattern Recognition, 2009, 2009. 1

[13] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009. 1

[14] Emily Denton, Sam Gross, and Rob Fergus. Semi-supervised learning with context-conditional generative adversarial networks. *arXiv preprint arXiv:1611.06430*, 2016. 2

[15] Terrance DeVries and Graham W Taylor. Improved regularization of convolutional neural networks with cutout. *arXiv preprint arXiv:1708.04552*, 2017. 1, 10

[16] Le Feng, Sheng Li, Zhenxing Qian, and Xinpeng Zhang. Unlabeled backdoor poisoning in semi-supervised learning. In *2022 IEEE International Conference on Multimedia and Expo (ICME)*, pages 1–6. IEEE, 2022. 3

[17] Yansong Gao, Change Xu, Derui Wang, Shiping Chen, Damith C Ranasinghe, and Surya Nepal. Strip: A defence against trojan attacks on deep neural networks. In *Proceedings of the 35th Annual Computer Security Applications Conference*, pages 113–125, 2019. 9, 14

[18] Tianyu Gu, Brendan Dolan-Gavitt, and Siddharth Garg. Badnets: Identifying vulnerabilities in the machine learning model supply chain. *arXiv preprint arXiv:1708.06733*, 2017. 1, 5

[19] Tianyu Gu, Kang Liu, Brendan Dolan-Gavitt, and Siddharth Garg. Badnets: Evaluating backdooring attacks on deep neural networks. *IEEE Access*, 7:47230–47244, 2019. 3, 11

[20] Alex Krizhevsky. Learning multiple layers of features from tiny images. Technical report, University of Toronto, 2009. 12

[21] Samuli Laine and Timo Aila. Temporal ensembling for semi-supervised learning. *arXiv preprint arXiv:1610.02242*, 2016. 2

[22] Dong-Hyun Lee et al. Pseudo-label: The simple and efficient semi-supervised learning method for deep neural networks. In *Workshop on challenges in representation learning, ICML*, volume 3, page 896, 2013. 1, 2

[23] Yuezun Li, Yiming Li, Baoyuan Wu, Longkang Li, Ran He, and Siwei Lyu. Invisible backdoor attack with sample-specific triggers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 16463–16472, 2021. 3

[24] Yige Li, Xixiang Lyu, Nodens Koren, Lingjuan Lyu, Bo Li, and Xingjun Ma. Neural attention distillation: Erasing back-

door triggers from deep neural networks. In *International Conference on Learning Representations*, 2020. 9, 14

[25] Yige Li, Xixiang Lyu, Nodens Koren, Lingjuan Lyu, Bo Li, and Xingjun Ma. Anti-backdoor learning: Training clean models on poisoned data. *Advances in Neural Information Processing Systems*, 34:14900–14912, 2021. 9, 15

[26] Yuncheng Li, Jianchao Yang, Yale Song, Liangliang Cao, Jiebo Luo, and Li-Jia Li. Learning from noisy labels with distillation. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1910–1918, 2017. 1

[27] Kang Liu, Brendan Dolan-Gavitt, and Siddharth Garg. Fine-pruning: Defending against backdooring attacks on deep neural networks. In *International Symposium on Research in Attacks, Intrusions, and Defenses*, pages 273–294. Springer, 2018. 14

[28] Yingqi Liu, Shiqing Ma, Yousra Aafer, Wen-Chuan Lee, Juan Zhai, Weihang Wang, and Xiangyu Zhang. Trojaning attack on neural networks. In *25th Annual Network And Distributed System Security Symposium (NDSS 2018)*. Internet Soc, 2018. 1

[29] Luis Muñoz-González, Battista Biggio, Ambra Demontis, Andrea Paudice, Vasin Wongrassamee, Emil C Lupu, and Fabio Roli. Towards poisoning of deep learning algorithms with back-gradient optimization. In *Proceedings of the 10th ACM Workshop on Artificial Intelligence and Security*, pages 27–38. ACM, 2017. 1, 2, 9

[30] Kevin P Murphy. *Machine learning: a probabilistic perspective*. MIT press, 2012. 1

[31] Nagarajan Natarajan, Inderjit S Dhillon, Pradeep K Ravikumar, and Ambuj Tewari. Learning with noisy labels. *Advances in neural information processing systems*, 26, 2013. 1

[32] Yuval Netzer, Tao Wang, Adam Coates, Alessandro Bissacco, Bo Wu, and Andrew Y. Ng. Reading digits in natural images with unsupervised feature learning. In *NIPS Workshop on Deep Learning and Unsupervised Feature Learning 2011*, 2011. 12

[33] Tuan Anh Nguyen and Anh Tran. Input-aware dynamic backdoor attack. *Advances in Neural Information Processing Systems*, 33:3454–3464, 2020. 3

[34] Aniruddha Saha, Akshayvarun Subramanya, and Hamed Pirsiavash. Hidden trigger backdoor attacks. In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, pages 11957–11965, 2020. 1, 5

[35] Mehdi Sajjadi, Mehran Javanmardi, and Tolga Tasdizen. Regularization with stochastic transformations and perturbations for deep semi-supervised learning. *Advances in neural information processing systems*, 29, 2016. 2

[36] Esha Sarkar, Hadjer Benkraouda, and Michail Maniatakos. Facehack: Triggering backdoored facial recognition systems using facial characteristics. *arXiv preprint arXiv:2006.11623*, 2020. 1, 3, 5

[37] Virat Shejwalkar and Amir Houmansadr. Manipulating the byzantine: Optimizing model poisoning attacks and defenses for federated learning. In *The Network and Distributed System Security Symposium (NDSS)*, 2021. 9

[38] V. Shejwalkar, A. Houmansadr, P. Kairouz, and D. Ramage. Back to the drawing board: A critical evaluation of poisoning attacks on production federated learning. In *2022 2022 IEEE Symposium on Security and Privacy (SP) (SP)*, pages 1117–1134, Los Alamitos, CA, USA, may 2022. IEEE Computer Society. 1, 2, 3, 9

[39] Connor Shorten and Taghi M Khoshgoftaar. A survey on image data augmentation for deep learning. *Journal of big data*, 6(1):1–48, 2019. 10

[40] Kihyuk Sohn, David Berthelot, Nicholas Carlini, Zizhao Zhang, Han Zhang, Colin A Raffel, Ekin Dogus Cubuk, Alexey Kurakin, and Chun-Liang Li. Fixmatch: Simplifying semi-supervised learning with consistency and confidence. *Advances in Neural Information Processing Systems*, 33:596–608, 2020. 1, 3, 5, 6, 12

[41] Kihyuk Sohn, Zizhao Zhang, Chun-Liang Li, Han Zhang, Chen-Yu Lee, and Tomas Pfister. A simple semi-supervised learning framework for object detection. *arXiv preprint arXiv:2005.04757*, 2020. 1

[42] Hossein Souri, Micah Goldblum, Liam Fowl, Rama Chellappa, and Tom Goldstein. Sleeper agent: Scalable hidden trigger backdoors for neural networks trained from scratch. *arXiv preprint arXiv:2106.08970*, 2021. 5

[43] Luke Taylor and Geoff Nitschke. Improving deep learning with generic data augmentation. In *2018 IEEE Symposium Series on Computational Intelligence (SSCI)*, pages 1542–1547. IEEE, 2018. 10

[44] Alexander Turner, Dimitris Tsipras, and Aleksander Madry. Label-consistent backdoor attacks. *arXiv preprint arXiv:1912.02771*, 2019. 1, 5, 9, 12

[45] Yidong Wang, Hao Chen, Yue Fan, Hao Wu, Bowen Zhang, Wenxin Hou, Yuhao Chen, and Jindong Wang. Torchssl: A pytorch-based toolbox for semi-supervised learning. https://github.com/TorchSSL/TorchSSL, 2021. [Online; accessed 03-July-2022]. 12

[46] Qizhe Xie, Zihang Dai, Eduard Hovy, Thang Luong, and Quoc Le. Unsupervised data augmentation for consistency training. *Advances in Neural Information Processing Systems*, 33:6256–6268, 2020. 1, 2, 5, 6

[47] Qizhe Xie, Minh-Thang Luong, Eduard Hovy, and Quoc V Le. Self-training with noisy student improves imagenet classification. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10687–10698, 2020. 1

[48] Zhicong Yan, Gaolei Li, Yuan TIan, Jun Wu, Shenghong Li, Mingzhe Chen, and H Vincent Poor. Dehib: Deep hidden backdoor attack on semi-supervised learning via adversarial perturbation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 10585–10593, 2021. 1, 3, 5, 9, 10

[49] Zhicong Yan, Jun Wu, Gaolei Li, Shenghong Li, and Mohsen Guizani. Deep neural backdoor in semi-supervised learning: threats and countermeasures. *IEEE Transactions on Information Forensics and Security*, 16:4827–4842, 2021. 3, 5

[50] Yi Zeng, Minzhou Pan, Hoang Anh Just, Lingjuan Lyu, Meikang Qiu, and Ruoxi Jia. Narcissus: A practical clean-label backdoor attack with limited information. *arXiv preprint arXiv:2204.05255*, 2022. 1, 2, 5, 7, 9, 11

[51] Yi Zeng, Won Park, Z Morley Mao, and Ruoxi Jia. Rethinking the backdoor attacks' triggers: A frequency perspective.

In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 16473–16481, 2021. 3, 9

[52] Bowen Zhang, Yidong Wang, Wenxin Hou, Hao Wu, Jindong Wang, Manabu Okumura, and Takahiro Shinozaki. Flexmatch: Boosting semi-supervised learning with curriculum pseudo labeling. *Advances in Neural Information Processing Systems*, 34, 2021. 1, 3, 5, 6

[53] Hongyi Zhang, Moustapha Cisse, Yann N Dauphin, and David Lopez-Paz. mixup: Beyond empirical risk minimization. In *International Conference on Learning Representations*, 2018. 2, 7

[54] Haoti Zhong, Cong Liao, Anna Cinzia Squicciarini, Sencun Zhu, and David Miller. Backdoor embedding in convolutional neural network models via invisible perturbation. In *Proceedings of the Tenth ACM Conference on Data and Application Security and Privacy*, pages 97–108, 2020. 5