EFFICIENT ANY-TARGET BACKDOOR ATTACK WITH PSEUDO POISONED SAMPLES

Author(s) Name(s)

Author Affiliation(s)

ABSTRACT

Deep neural networks present their potential vulnerabilities under backdoor attacks. They have a satisfactory performance for benign users with clean inputs but will get malicious outputs when samples are attached with the backdoor trigger. Currently, most of the backdoor attacks target only one single class, making them not robust to defenses against this characteristic. In this work, we explore an invisible and input-dependent any-target attack targeting all the labels simultaneously. Specifically, we train the attacked model together with the steganography model by encoding the one-hot encodings into the images. The novel pseudo poisoned samples are proposed to improve the effectiveness of our attack. Experimental results show that our method is both efficient and effective on several datasets and robust to existing defenses.

***Index Terms*—** Backdoor Attack, Any-Target Attack, Steganography, Pseudo Poisoned Samples

1. INTRODUCTION

Deep neural networks have been applied in a variety of applications due to their powerful ability of data representation. However, the rigorous necessity of massive training data, expensive computing resources and long training time discourage the entities that lack these conditions. As an alternative, they may turn to third-parties for training their models or using the pre-trained ones directly, which brings new surface of potential security threats.

BadNets [1] is the first proposed backdoor attack on neural networks, where the attacker poisons a portion of training samples by attaching a pre-defined pattern (termed trigger) on them and changes the corresponding labels to a target one. Models trained on this partially modified training set will be embedded with hidden backdoor, predicting the target label when the trigger shows up while behaving normally otherwise. Studies on backdoor attack usually focus on its effectiveness or stealthiness, where the former tries to find out better triggers to increase the attack success rate [2, 3]. As most attacks rely on triggers that are visible and static, making them easily noticeable visually or detectable algorithmically, many invisible [3, 4], dynamic [5, 6] or input-dependent [5, 7, 8] triggers have be proposed continually. What's more, backdoor attack without label modification, which is called clean-label attack [9, 10, 11], can also enhance the stealthiness.

From the perspective of the target labels, we argue that targeting multiple labels simultaneously will make the attack harder to detect. There is a kind of multi-target attack named all-target attack in current studies [1, 5], where the attacker aims to misclassify poisoned samples as their next labels, i.e. classify samples in class c as class $c + 1$. Here we try a more aggressive attack, any-target attack, where the attacker generates poisoned samples of any target classes and leads the attacked model to predict them as the specified classes. Some existing single-target attacks [1, 12, 13] can be easily extended to our any-target attacks by mapping distinct triggers to different labels. But the process is too troublesome and the extended attacks are inefficient and ineffective. A few attacks [6, 14] are applicable to the any-target attack paradigm, but they are visible and input-independent.

Inspired by image steganography [15, 16] and steganography based backdoor attacks [7, 17], we propose to train the model to be attacked together with the steganography model by replacing its decoder with the former. The target label is specified by its corresponding one-hot encoding which is then encoded into the benign input image to produce a poisoned one. As thus, the tedious process of selecting distinct triggers in any-target attack can be efficiently implemented by just offering the one-hot encodings. What's more, the poisoned images are invisible and input-dependent owing to the encoder of steganography. In the end, to encourage the attacked model to better learn the mapping from the triggers to the target labels, we propose the novel pseudo poisoned samples which are also poisoned samples generated by the encoder but with their labels unchanged, ensuring the effectiveness of the any-target attack.

We evaluate our attack on four datasets MNIST, CIFAR-10, GTSRB and CelebA. Experiments show that on every dataset we achieve high attack success rate of over 99% and similar benign accuracy with the benign model, even under a rather low attack ratio. The triggers added to the poisoned images are visually imperceptible. The proposed attack can also bypass several typical defense methods. Ablation studies reveal the importance of the novel pseudo poisoned samples to the success of our attack whose potential of attacking larger number of classes has been verified as well.

Thanks to XYZ agency for funding.

Our main contributions are listed as follows:

(1) We explore an invisible and input-dependent any-target backdoor attack to make it more stealthy to human inspection or algorithmic detection.

(2) We propose novel pseudo poisoned samples to improve the performance of our attack.

(3) Extensive experiments demonstrate the effectiveness and efficiency of the proposed method and its robustness to several defense methods.

2. THE PROPOSED ATTACK

2.1. Problem Definition

Definition of Backdoor Attack. Given a clean training set $\mathcal{D} = \{(x, y)\}$, where x is an input image and y is the corresponding label, an attacker applies a backdoor injection function $\mathcal{B}$ and a label mapping function $\mathcal{M}$ on part of $\mathcal{D}$ to generate a poisoned training set $\mathcal{T} = \{(x_t, y_t)\} = \{(\mathcal{B}(x), \mathcal{M}(y))|(x, y) \in \mathcal{D}', \mathcal{D}' \subset \mathcal{D}\}$. For example, BadNets [1] uses $\mathcal{B}$ to add a static small patch on the fixed position of an image as follows:

$$\mathcal{B}(x) = x \odot (1 - m) + p \odot m, \tag{1}$$

where p is a pattern, m a blending mask and $\odot$ pixel-wise multiplication. In almost all the existing backdoor method, $\mathcal{M}$ is a constant function $\mathcal{M}(y) = \hat{y}$ (single-target attack), where $\hat{y}$ is the target label, or a shifting function $\mathcal{M}(y) = (y+1) \bmod C$ (all-target attack), where C is the class number. The attack ratio is defined as the ratio of the poisoned sample volume to the training sample volume, i.e. $\alpha = |\mathcal{T}|/|\mathcal{D}|$. After poisoning the training set, the attacker injects a backdoor into the classification model by training it over $\mathcal{T}$ and the non-poisoned part of $\mathcal{D}$, i.e. $\mathcal{T} \cup \mathcal{D} \backslash \mathcal{D}'$. The goal of the attacker is to make targeted classification if the backdoor trigger is present at test-time examples while without harming its classification accuracy on benign test samples.

Definition of Any-Target Attack. We refer to our attack paradigm as any-target attack to distinguish it from the all-target attack. Different from the definition of previous attack, the backdoor injection function $\mathcal{B}$ of any-target attack takes as input an image as well as a target class, then it generates a poisoned sample via $x_t = \mathcal{B}(x, t)$. While the mapping function $\mathcal{M}$ just outputs the target class t used in the former generating process. By encoding the class information into the image, there will be some perturbation in part of the pixel values. We can treat it as the trigger added to the sample for backdoor effect. It should be noted that the perturbed pixel number or intensity must be small enough for invisibility consideration.

Attacker's Capacities. We consider one commonly used threat model by existing backdoor attack studies. The attacker has full knowledge of the training dataset and training algorithm. He can modify the training data as he want and intervene the training process for desired purpose. The infected model is then distributed for users to download.

Attacker's Goals. After injecting the backdoor, it is expected that two main goals are achieved. To be specific, on clean images, the attacked model shows similar classification performance with that not attacked, while on the generated examples by $\mathcal{B}$, the model predicts the target class that is encoded in the input with high probability.

2.2. Proposed Method

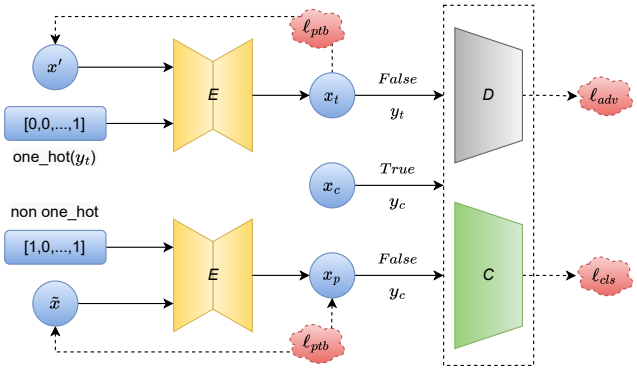


Fig. 1. The overall workflow of our proposed method. Three modes of the training process are illustrated. The labels after x are ground truth for D and C respectively.

Overview. The overall framework of our proposed attack is illustrated in Fig. 1. We take the image steganography model Hidden [15] as our base. Specifically, Hidden comprises four main components: an encoder, a noise layer, a decoder and a discriminator. We keep its encoder and discriminator as our *encoder* E and *discriminator* D, while replace its decoder with the *classifier* C to be attacked, as our purpose is to classify the encoded image as the target rather than recover the encoded message from it.

Given a dataset $\mathcal{D}$ with N samples, we randomly take α of them as the poisoned set $\mathcal{D}'$. The encoder E receives an image x' from $\mathcal{D}'$ and a random target label y_t and produces a poisoned image x_t which is encoded with y_t. Note that any vector can be used to represent the target label if only they are one-to-one mapped. In our method, we choose the one-hot encoding for simplicity. The classifier C predicts an image x_c from $\mathcal{D}$ as its ground truth class y_c and the poisoned image x_t as its encoded class y_t. At the same time, the discriminator D tries to predicts the probability that the input image, either x_c or x_t, is not poisoned.

The poisoned image should have no obvious visual difference with the corresponding benign one. We limit the variation of the label encoding process with a *perturbation loss* ℓ_{ptb}, the ℓ_2 distance between x' and x_t, and a *adversarial loss* ℓ_{adv}, the binary cross entropy (BCE) that aims to fool the dis-

criminator. For the attacked classifier, we impose a *classification loss* ℓ_{cls} using the cross entropy (CE) to ensure the expected prediction. The discriminator incurs *adversarial loss* from both x' and x_t, which reflects the ability to distinguish the poisoned image.

Pseudo Poisoned Samples. In order to make the attack more effective and more efficient, we introduce the *pseudo poisoned samples*. A pseudo poisoned sample is also a poisoned image generated by the encoder via encoding into it some vector dubbed *pseudo encoding*. As the one-hot encodings are taken for representing the target labels, we choose one of the rest non one-hot encodings as the input vector of E, that is, there are zero or more than one 1 in it. It should be noted that the length of the pseudo encoding equals to a one-hot encoding's. The difference between poisoned samples and pseudo poisoned samples lies in their ground truth labels.

We first randomly take the same amount, i.e. ratio α, of samples from $\mathcal{D}$ as the pseudo poisoned set $\tilde{\mathcal{D}}$. Then, an image $\tilde{x}$ from $\tilde{\mathcal{D}}$ is sent to E in company with a pseudo encoding, resulting a pseudo poisoned sample x_p. When calculating the classification loss, the ground truth label remains unchanged. Other parts of the workflow of $\tilde{x}$ are identical with that of x'. After introducing the pseudo poisoned samples, we improve the ability of the attacked model to distinguish poisoned samples from others. As is shown in Sec. 3.2, the attack performance improves significantly.

Brief Summary. The three modes of the training process are summarized as: (1) **clean mode**. The classifier C has to correctly recognize x_c. (2) **attack mode**. When x_t is encoded with a target label, the attack is activated, making C output wrong prediction y_t. (3) **pseudo attack mode**. Although encoded with a similar vector, x_p can still not fool C. The encoder E and classifier C are jointly trained by minimizing the loss $\mathcal{L}_1$ in Eq. (2) through stochastic gradient descent. And the discriminator D is trained likewise by minimizing the loss $\mathcal{L}_2$ in Eq. (3) as follows:

$$\begin{aligned}\mathcal{L}_1 = &\mathbb{E}_{(x,y)\in\mathcal{D}}\left[\lambda_{cls}\ell_{cls}(C(x),y)\right] \\ &+ \mathbb{E}_{(x,y)\in\mathcal{D}'\cup\tilde{\mathcal{D}}}\left[\lambda_{cls}\ell_{cls}(C(E(x)),\mathcal{M}(y))\right. \\ &\left.+ \lambda_{ptb}\ell_{ptb}(x,E(x)) + \lambda_{adv}\ell_{adv}(D(E(x)),1)\right], \quad (2)\\ \mathcal{L}_2 = &\mathbb{E}_{x\in\mathcal{D}'\cup\tilde{\mathcal{D}}}\left[\ell_{adv}(D(x),1) + \ell_{adv}(D(E(x)),0)\right], \quad (3)\end{aligned}$$

where λ_{cls}, λ_{ptb} and λ_{adv} are hyperparameters that balance each loss. Note that the two items in $\mathcal{L}_2$ share equal weight. We alternately optimize the two parts mentioned above.

3. EXPERIMENTAL RESULTS

3.1. Experimental Settings

Datasets and Models. We conduct our attack experiments on four datasets: MNIST [18], CIFAR-10 [19], GTSRB [20] and CelebA [21]. The classifier for MNIST is a self-defined neural network following previous studies [5]. While for the other three datasets, we use Pre-activation Resnet-18 [22]. The encoder and discriminator are kept consistent with Hidden [15] except for the input layer. Moreover, to verify the effectiveness on larger number of classes, we randomly select a subset of 200 classes from ImageNet [23].

Training Setup. We adopt the SGD optimizer to train all the modules, and the initial learning rate is set as 0.01, which drops to 0.1 times after every 100 epochs. The batch size and maximum epoch are 128 and 300 respectively. We set the attack ratio α as 10% and decrease it by 2% each time for comparison until 2%. The hyperparameters λ_{cls}, λ_{ptb} and λ_{adv} are set to 1, 0.7 and 0.001. All the experiments are conducted on NVIDIA GeForce RTX 3090 GPUs.

Baseline Selection. We extend BadNets [1], Blended [24] and ISSBA [7] to any-target attack. To be specific, N distinct triggers are mapped to all N classes. For example, we elaborately design different patterns for BadNets, select diverse images to blend for Blended and choose various strings to encode for ISSBA. We compare our method with those attacks as well as the benign model without backdoor injection.

Evaluation Metrics. We use benign accuracy (ACC) and attack success rate (ASR) for evaluation. ACC is defined as the accuracy of testing on benign samples while ASR is that on poisoned samples.

3.2. Main Results

Performance under Different Attack Ratios. Tab. 1 reports the comparison of different attacks. As can be seen, under $\alpha = 0.1$, our attack can achieve a high ASR approaching to 100% on all four datasets, better than the others, while keeping ACC very close to the benign model. On the contrary, BadNets or Blended can only have comparable high ASR on part of the datasets. We also experiment on the extended ISSBA, but result a failure attack where ASR equals to the probability of random guess, i.e. the reciprocal of the number of classes. Besides, even under more lower α, the ASR of our attack still stays at a high level of over 99%, while that of other attacks drops significantly, revealing the effectiveness and efficiency of ours. Note that there is slight reduction on the ACC of our method when α decreases to 0.2, which means it focuses more on learning the poisoned samples.

(a) Benign (b) Poisoned (c) Residual×10

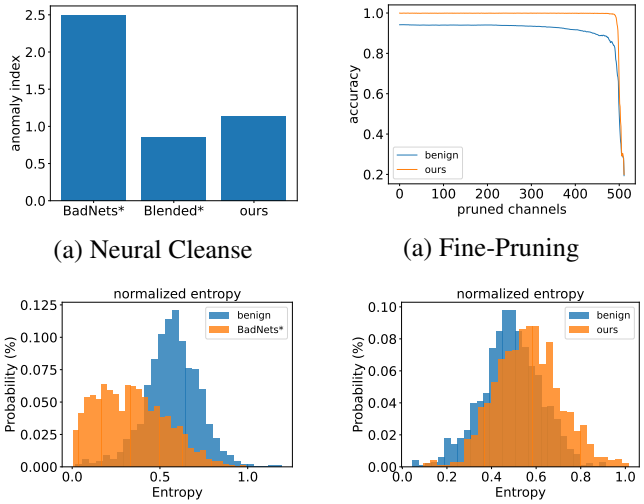
Fig. 2. Example of placing a figure with experimental results.

Visualization of Poisoned Images. Fig. 2 presents an example of the poisoned images generated by our method. The poisoned image brings satisfactory visual invisibility, making

Table 1. The table is inserted here. (%)

attack ↓	$\alpha \rightarrow$	MNIST			CIFAR10			GTSRB			CelebA		
		0.1	0.06	0.02	0.1	0.06	0.02	0.1	0.06	0.02	0.1	0.06	0.02
benign	ACC		99.58			94.74			99.20			80.26	
BadNets*	ACC	99.47	99.40	99.44	93.52	93.82	94.04	**98.64**	**99.03**	98.60	**79.94**	79.29	79.09
	ASR	95.52	95.18	89.78	48.84	49.64	48.35	94.94	93.03	59.77	77.41	78.55	77.80
Blended*	ACC	99.59	99.60	**99.50**	93.68	94.00	**94.21**	98.62	98.73	98.95	79.91	**79.57**	**79.27**
	ASR	99.97	99.92	97.98	93.86	88.25	59.57	90.58	82.17	56.00	99.66	99.11	97.12
ours	ACC	**99.60**	**99.65**	99.47	**94.18**	**94.02**	92.21	98.54	98.91	**99.04**	79.44	79.31	78.76
	ASR	**100.0**	**100.0**	**100.0**	**99.97**	**99.84**	**99.45**	**99.26**	**99.01**	**99.81**	**99.70**	**99.89**	**99.58**

it look natural to human inspection and reflecting the stealthiness of our attack.



(a) Neural Cleanse (a) Fine-Pruning

(c) Strip

Fig. 3. Example of placing a figure with experimental results.

Resistant to Backdoor Defenses. We test our approach against three defenses: Neural Cleanse [25] (NC), Fine-Pruning [26] and Strip [27]. NC reverse engineers candidate triggers for each class and finds out significantly small one as the target. It calculates an *anomaly index* which indicates an anomaly when greater than 2. As shown in Fig. 3(a), our attack successfully bypass NC. When pruning the channels on the penultimate layer increasingly, our attack retains its high ASR even when almost all the channels are pruned as in Fig. 3(b), showing the resistance to pruning-based defense. Strip determines candidate poisoned samples by calculating the entropy of the average prediction on the samples generated by imposing various pre-prepared clean samples on the testing samples. Poisoned samples are expected to have relative smaller entropy. But in our attack, as shown in Fig. 3(c), the entropy is close to, even slightly larger than that in the benign model. All the results on Fig. 3 are on CIFAR-10 and results on other datasets are similar.

Table 2. PPS is xxx, ITP is yyy. (%)

attack ↓	CIFAR10		GTSRB		CelebA	
	ACC	ASR	ACC	ASR	ACC	ASR
w/ both	**94.18**	**99.97**	**98.54**	**99.26**	**79.44**	**99.70**
w/o PPS	88.38	76.68	18.27	97.85	75.17	97.88
w/o ITP	90.40	48.17	98.23	55.71	76.21	96.11

Necessity of Pseudo Poisoned Samples. We carry out a variation of our attack, which consists of only the first two modes in the training process, that is, without pseudo attack mode. As we can see in Tab. 2, after removing the pseudo poisoned samples, both ACC and ASR drop dramatically, which demonstrates the necessity of them.

Necessity of Intervening Training Process. As our trained encoder E can successfully encode one-hot encoding into images, we are interesting to find out whether we can use the poisoned samples generated by E to attack the classifier without intervening the training process, as the baselines do. We show in Tab. 2 that its ASR is far inferior to our method's, but is better than that of the extended ISSBA, which equals to random guessing probability.

Potential of Target Class Numbers. We perform several attacks on a subset of 200 classes from ImageNet for every additional 20 classes. Experimental results indicate that the ASR drops a little as the class number increases when it is larger than 140, but still reaches about 90% when it is 200.

4. CONCLUSION

5. REFERENCES

[1] Tianyu Gu, Brendan Dolan-Gavitt, and Siddharth Garg, "Badnets: Identifying vulnerabilities in the ma-

chine learning model supply chain," *arXiv preprint arXiv:1708.06733*, 2017.

[2] Yingqi Liu, Shiqing Ma, Yousra Aafer, Wen-Chuan Lee, Juan Zhai, Weihang Wang, and Xiangyu Zhang, "Trojaning attack on neural networks," 2017.

[3] Khoa Doan, Yingjie Lao, Weijie Zhao, and Ping Li, "Lira: Learnable, imperceptible and robust backdoor attacks," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, pp. 11966–11976.

[4] Erwin Quiring and Konrad Rieck, "Backdooring and poisoning neural networks with image-scaling attacks," in *2020 IEEE Security and Privacy Workshops (SPW)*. IEEE, 2020, pp. 41–47.

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

[6] Ahmed Salem, Rui Wen, Michael Backes, Shiqing Ma, and Yang Zhang, "Dynamic backdoor attacks against machine learning models," in *2022 IEEE 7th European Symposium on Security and Privacy (EuroS&P)*. IEEE, 2022, pp. 703–718.

[7] 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*, 2021, pp. 16463–16472.

[8] Le Feng, Sheng Li, Zhenxing Qian, and Xinpeng Zhang, "Stealthy backdoor attack with adversarial training," in *ICASSP 2022-2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2022, pp. 2969–2973.

[9] Ali Shafahi, W Ronny Huang, Mahyar Najibi, Octavian Suciu, Christoph Studer, Tudor Dumitras, and Tom Goldstein, "Poison frogs! targeted clean-label poisoning attacks on neural networks," *Advances in neural information processing systems*, vol. 31, 2018.

[10] Aniruddha Saha, Akshayvarun Subramanya, and Hamed Pirsiavash, "Hidden trigger backdoor attacks," in *Proceedings of the AAAI conference on artificial intelligence*, 2020, vol. 34, pp. 11957–11965.

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

[12] Xueluan Gong, Yanjiao Chen, Qian Wang, Huayang Huang, Lingshuo Meng, Chao Shen, and Qian Zhang, "Defense-resistant backdoor attacks against deep neural networks in outsourced cloud environment," *IEEE Journal on Selected Areas in Communications*, vol. 39, no. 8, pp. 2617–2631, 2021.

[13] Nan Zhong, Zhenxing Qian, and Xinpeng Zhang, "Imperceptible backdoor attack: From input space to feature representation," *arXiv preprint arXiv:2205.03190*, 2022.

[14] Xinzhe Zhou, Wenhao Jiang, Sheng Qi, and Yadong Mu, "Multi-target invisibly trojaned networks for visual recognition and detection.," in *IJCAI*, 2021, pp. 3462–3469.

[15] Jiren Zhu, Russell Kaplan, Justin Johnson, and Li Fei-Fei, "Hidden: Hiding data with deep networks," in *Proceedings of the European conference on computer vision (ECCV)*, 2018, pp. 657–672.

[16] Matthew Tancik, Ben Mildenhall, and Ren Ng, "Stegastamp: Invisible hyperlinks in physical photographs," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 2117–2126.

[17] Shaofeng Li, Minhui Xue, Benjamin Zi Hao Zhao, Haojin Zhu, and Xinpeng Zhang, "Invisible backdoor attacks on deep neural networks via steganography and regularization," *IEEE Transactions on Dependable and Secure Computing*, vol. 18, no. 5, pp. 2088–2105, 2020.

[18] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.

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

[20] Johannes Stallkamp, Marc Schlipsing, Jan Salmen, and Christian Igel, "Man vs. computer: Benchmarking machine learning algorithms for traffic sign recognition," *Neural networks*, vol. 32, pp. 323–332, 2012.

[21] Ziwei Liu, Ping Luo, Xiaogang Wang, and Xiaoou Tang, "Deep learning face attributes in the wild," in *Proceedings of the IEEE international conference on computer vision*, 2015, pp. 3730–3738.

[22] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun, "Identity mappings in deep residual networks," in *European conference on computer vision*. Springer, 2016, pp. 630–645.

[23] 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*. Ieee, 2009, pp. 248–255.

[24] 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.

[25] 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 *2019 IEEE Symposium on Security and Privacy (SP)*. IEEE, 2019, pp. 707–723.

[26] 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*. Springer, 2018, pp. 273–294.

[27] 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*, 2019, pp. 113–125.