EXPLOITING BACKDOOR TRIGGER TOWARDS UNLEARNABLE EXAMPLES

Chun Li[†‡] *Yuanchen Xun*[†‡] *Zhong Li*[†‡] *Minxue Pan*[†‡*] *Xuandong Li*[†§]

[†] State Key Laboratory for Novel Software Technology, Nanjing University, China
[‡] Software Institute, Nanjing University, China
[§] School of Computer Science, Nanjing University, China

ABSTRACT

Unlearnable examples (UEs) aim to protect personal privacy from abuse in model training. Existing work primarily generates UEs by crafting imperceptible perturbations through search or optimization and introducing them to clean samples. Despite considerable progress in UE methods, their applicability in real-world scenarios and effectiveness under different defense strategies remains limited. In this paper, we reveal the connections between poisoning-based backdoor attacks and UEs from the perspective of shortcut learning. Specifically, both the perturbations in UEs and backdoor triggers are easy-to-learn features, and thus models trained on samples with them will ignore other important semantic features and perform poorly on clean data. Motivated by the observation, we propose TRIGGERUE, which utilizes the backdoor triggers to craft effective UEs for the first time. Our experimental results show that our method significantly outperforms existing state-of-the-art UE methods. Code: https://github.com/pppppkun/TriggerUE.

Index Terms— Unlearnable Examples, Backdoor Attacks

1. INTRODUCTION

A key prerequisite for training a high-performance deep neural network is a large-scale, high-quality training dataset [1, 2]. To achieve this, individuals or organizations tend to use web crawlers to scrape data from the internet [3, 4]. However, unauthorized collection or illegal purposes may raise serious privacy concerns. For example, a company has been continuously collecting facial images from the internet for training commercial facial recognition models [5]. To protect personal data from abuse without affecting user experience, researchers have proposed utilizing unlearnable example (UE) methods [6, 7, 8, 9, 10] that introduce imperceptible perturbations to the clean data to craft UEs, ensuring that unauthorized models trained on UEs perform poorly on clean data distributions.

Despite the significant progress that has been made, existing UE methods still exhibit certain limitations. Firstly, they assume the availability of an entire dataset or a known number of classes [6, 11, 12, 10, 9], which is challenged in real-world scenarios where data is dynamic or even streaming [13, 6]. Secondly, adversarial training [7] and strong data augmentation [14, 15, 16] have been demonstrated to defend against current UE methods [12, 9, 17]. These approaches restore the test accuracy decreased by UEs, thereby rendering privacy protection less effective.

In this paper, we reveal the connections between poisoning-based backdoor attacks and UEs from the perspective of shortcut learning. Specifically, shortcut learning [18, 19, 20] suggests that models tend to ignore semantic features and focus on easy-to-learn features that are sufficient for distinguishing examples from different classes. From this perspective, both the perturbations in UEs and backdoor triggers can be treated as the easy-to-learn semantic features that can cause the models to primarily rely on them for prediction while ignoring other important semantic features [9, 8, 21, 22, 23]. Based on this observation, we propose TRIGGERUE, which crafts UEs by injecting backdoor triggers. Our TRIGGERUE consists of two main steps, including **feature elimination** and **trigger injection**. Specifically, TRIGGERUE first generates adversarial noises that aim to eliminate the semantic features of clean samples. Then, we generate class-wise triggers and inject the same trigger for the samples of the same class. By injecting the class-wise triggers, the models would treat these triggers as classification shortcuts and ignore other semantic features, thereby achieving poor generalization and personal privacy protection. We evaluated the effectiveness of our approach and compared it to baselines across various datasets, defense strategies, and adaptive defenses. Our experimental results demonstrate that TRIGGERUE achieves better performances across different experimental settings.

[*]Corresponding author (mxp@nju.edu.com).

2. BACKDOOR TRIGGER FOR UNLEARNABLE EXAMPLES

2.1. Problem Definition

Threat Model. Following previous work [6, 7, 11], we focus on serving as a protector, preventing unauthorized use of protected data for model training, thereby safeguarding personal privacy. We highlight that our goal is to provide protection in practical, real-world scenarios where data is growing in real-time. For example, on social media, users continuously upload new data.

Protector's and Infringer's Goal. For the protector, the objective is to release a new dataset without affecting user experience, ensuring that unauthorized models trained on this dataset perform poorly on clean data distributions. We consider an infringer as an unauthorized model trainer. The goal of the infringer is to produce a model that generalizes well on clean data distribution via training on the protected dataset.

Protector's and Infringer's Capability. The protector has full control over all data requiring protection. The protector does not have any knowledge of the training process and cannot control it. The infringer has access to the released datasets and has full control of the training process of the model. However, the infringer is unaware of the distribution or the proportion of unlearnable examples within the given training dataset.

Defining Unlearnable Examples. We formulate the problem in the context of image classification with DNNs. Let $\mathcal{D}_c = \{(\boldsymbol{x}_i, y_i)\}_{i=1}^n$ denotes the clean training dataset contains n clean examples, where $\boldsymbol{x}_i \in \mathcal{X} \subset \mathbb{R}^d$ is i-th image, $y_i \in \mathcal{Y} = \{1, \ldots, K\}$ is its label, and K is the number of classes. We denote the clean test dataset as $\mathcal{D}_t$,

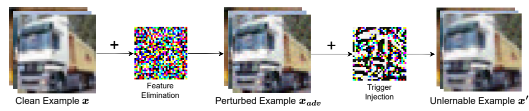
+
Feature Elimination
+
Trigger Injection
Clean Example $\boldsymbol{x}$
Perturbed Example $\boldsymbol{x}_{adv}$
Unlernable Example $\boldsymbol{x}'$

Fig. 1: The process of generating UEs by TRIGGERUE

where $\mathcal{D}_t$ shares the same distribution as $\mathcal{D}_c$. The problem targeted in this paper is to protect the data from unauthorized training by perturbing $\mathcal{D}_c$ into an unlearnable dataset as $\mathcal{D}_u = \{(\boldsymbol{x}_i + \delta_i, y_i)\}_{i=1}^{n}$, where $\delta_i \in \Delta \subset \mathbb{R}^d$ is the perturbation. The Δ is the perturbation set of $\mathcal{D}_c$ and usually bounded by $\|\delta\|_p < \epsilon$, where $\|\cdot\|_p$ is the ℓ_p norm and ϵ is set sufficiently small such that it does not affect the utility of the example. To obtain the perturbations, we often need to optimize the following problem:

$$\begin{aligned} &\max_{\delta \in \Delta} \mathbb{E}_{(\boldsymbol{x},y)\sim D_t}[\mathcal{L}(f(\boldsymbol{x};\theta(\delta)), y)] \\ \text{s.t. } \; &\theta(\delta) = \arg\min_{\theta} \mathbb{E}_{(\boldsymbol{x},y)\sim D_c}[\mathcal{L}(f(\boldsymbol{x}+\delta;\theta), y)] \end{aligned} \tag{1}$$

where $\mathcal{L}$ is classification loss such as cross-entropy loss, f is the model infringer intend to train and θ is parameters of f. Intuitively, Equation 1 aims to find a perturbation set such that a fully trained model on the unlearnable examples $\boldsymbol{x} + \delta$ exhibits the worst performance on a clean test set $\mathcal{D}_t$.

2.2. Observations

Our key observation to craft unlearnable examples is that poisoning-based backdoor attacks and UEs share similar properties from the perspective of shortcut learning. Shortcut learning suggests that models tend to focus on easy-to-learn features that are sufficient for distinguishing examples, no matter whether it is semantic or not [19, 20, 18]. From this perspective, the perturbations in UEs can be treated as easy-to-learn features that are able to establish classification shortcuts from perturbations to labels, thus preventing the models from learning other semantic features [9, 8, 13]. Similarly, models trained on poisoned samples (i.e., samples with injected triggers) quickly learn stable correlations between triggers and target labels. This suggests that backdoor triggers are also easy-to-learn features that can create classification shortcuts [21, 24]. Based on this observation, if we inject a class-wise trigger into all samples of each class, the model will explicitly learn the connection between the trigger and the label while ignoring other semantic features, thus reducing generalization to the clean test set. This indicates that utilizing triggers to craft UEs is feasible. Furthermore, the generation and injection of triggers typically do not necessitate using surrogate models, nor do they involve solving optimization or search tasks [25, 26, 27, 28]. This makes the generation of UEs more efficient and may achieve better effectiveness across different model architectures. Moreover, injecting triggers into a given sample does not require access to other samples, which eliminates the assumption of an entire dataset or a known number of classes and enhances flexibility in scenarios with growing data. Employing triggers, as demonstrated in prior research [29, 21, 30, 31], can also better resist adversarial training and strong data augmentation.

2.3. Proposed Method

Based on these observations, we propose a novel unlearnable example generation approach, named TRIGGERUE. TRIGGERUE consists of two main steps: (1) the **feature elimination** step aims to eliminate the easy-to-learn semantic features of the clean samples, and (2) the **trigger injection** step focuses on injecting class-wise backdoor triggers into samples to produce unlearnable examples. Figure 1 overviews the high-level of TRIGGERUE. TRIGGERUE takes as input a clean example $(\boldsymbol{x}, y)$ and outputs an unlearnable example $(\boldsymbol{x}', y)$ for this clean example. We will elaborate on each step in the following.

2.3.1. *Feature Elimination.*

Before injecting triggers into samples, we first introduce adversarial noises to perturb their semantic features. The reason is that the model trained on samples with triggers can still learn semantic features that are useful for classification, leading to sub-optimal unlearnability. This is potentially because clean samples contain some easy-to-learn semantic features that are useful for classification, as shown in prior research [32, 33]. To eliminate these features, we generate the adversarial noises and inject them into the clean samples, as the process of optimizing noise involves eliminating features in the samples that facilitate classification [34]. Specifically, we optimize the adversarial example generation objective to craft the adversarial noises [34, 6]:

$$\max_{\delta_i} [\mathcal{L}(F(\boldsymbol{x}_i + \delta_i; \theta^*), y)] \;\; \text{s.t. } \; \|\delta\|_p < \epsilon \tag{2}$$

where F denote a model and θ^* are the parameters of F, $\mathcal{L}$ is the commonly used cross entropy loss. In our threat model, the protector does not have any knowledge of the model F trained by the infringer, making it challenging to directly optimize Eq. 2. To address this challenge, we leverage the transferability of adversarial noises to generate transferable adversarial noises on a surrogate model. We adopt OpenAI's large-scale vision-and-language pre-trained model CLIP [35] as our surrogate model because it has been demonstrated to be effective in generating highly transferable adversarial noise [36]. To solve Eq. 2, as suggested in prior work [6, 7], we employ the first-order optimization method Projected Gradient Descent (PGD) [7]. More specifically, Eq. 2 is solved by PGD as:

$$\delta^{(t+1)} = \Pi_\epsilon \left(\delta^{(t)} + \alpha \cdot \text{sign}(\nabla_\delta \mathcal{L}(F(\boldsymbol{x} + \delta^{(t)}; \theta^*), y)) \right), \tag{3}$$

where t is the current step (T steps in total), $\nabla_\delta \mathcal{L}(F(\boldsymbol{x}+\delta^{(t)}; \theta^*), y)$ is the gradient of the loss, Π_ϵ is a projection function that clips the noise by $\|\delta\| < \epsilon$, and α is the step size. Note that because we use CLIP as the surrogate model, y is the natural language description of the sample in Eq. 3. We directly use the class name to which the sample belongs as y.

2.3.2. *Trigger Injection*

In TRIGGERUE, we inject class-wise triggers into samples to produce unlearnable examples. As mentioned earlier, backdoor triggers serve as easy-to-learn features to prevent the models from learning other semantic features. As such, by injecting class-wise triggers, the model would learn to distinguish between different classes solely based on the features of backdoor triggers, thus resulting in poor generalization performance on the clean data distributions. Furthermore, we require the triggers to be class-wise because if two classes use the same trigger, the model would be compelled to identify and learn additional semantic features to distinguish between these classes, thereby counteracting the effectiveness of UEs.

We employ the warping-based backdoor attack WaNet [27] to generate class-wise triggers. The reason why we adopt WaNet is that WaNet can easily generate corresponding triggers for new classes dynamically produced in real-world scenarios. The capability of WaNet originates from it generates triggers through a randomly sampled

Table 1: Test accuracy (%) of ResNet-18 trained on clean data vs. UEs by different baselines.

	SVHN	CIFAR-10	CIFAR-100	ImageNet
Clean	95.94	95.1	75.51	53.15
TAP	65.21	17.07	12.76	27.59
EM	10.32	12.31	7.2	4.78
REM	13.34	14.29	3.43	7.23
EntF	82.27	84.85	50.06	34.22
LSP	8.23	14.75	8.78	3.44
OPS	9.44	13.56	11.69	6.54
AR	6.742	12.54	5.12	1.45
TRIGGERUE	**6.108**	**9.46**	**2.21**	**0.78**

uniform grid, enabling it to produce an arbitrary number of triggers. More specifically, we follow the default implementation of WaNet to initialize and inject triggers. We first sample a uniform grid of size $k \times k \times 2$, then normalize it and multiply it by a warping strength s. This grid is then upsampled to match the size of the image to serve as the trigger. Once the trigger is obtained, we inject it into the original image through the warping function $\mathcal{W}$ used in WaNet, which is implemented by the bi-linear interpolation function.

3. EXPERIMENT

3.1. Experiment Settings

Datasets. We extensively evaluate the effectiveness of TRIGGERUE on SVHN [37], CIFAR10 [38], CIFAR100 [38], and the Tiny ImageNet [39]. These datasets are the most widely used datasets in previous work [6, 7, 11, 12, 9].

Implementation. In feature elimination, we employed CLIP (ResNet50 as the backbone) for our surrogate model. When using PGD (Eq. 3) to solve Eq. 2 for adversarial noises, we use ℓ_∞ bound to $\epsilon = 8/255$, $\alpha = 0.05$, and $T = 20$. In trigger injection, we use uniform grid size $k = 4$ and warping strength $s = 0.5$ to initialize the trigger as recommended in WaNet [27].

Baselines. Following previous work [9, 12, 11, 10], we selected 7 representative UEs as our baselines, which are Target Adversarial Poisoning (TAP) [6], Error-Minimizing (EM) [7], Robust Error-Minimizing (REM) [11], Entangled Features (EntF) [12], Linear Synthetic Perturbation (LSP) [8], One-Pixel Shortcut (OPS) [9], and Autoregressive Perturbations (AR) [10]. For the studied baselines, we directly used their official implementations along with the corresponding parameter configurations.

Generation and evaluation. In all experiments, we followed the same evaluation process as previous works [6, 7]: we trained the model on the unlearnable dataset and evaluated the trained model on the clean test set to assess its performance. Unless specifically stated, standard data augmentation techniques were used when training models, and experiments were conducted on CIFAR-10 using ResNet-18 [40].

Metric. In line with prior work [6, 7], we evaluate the effectiveness of UEs by examining the test accuracy of the model trained on them. The lower the test accuracy, the better the effectiveness.

3.2. Effectiveness

This section presents the experimental results to demonstrate the effectiveness of our proposed TRIGGERUE. In Table 1, each row represents the test accuracy of the same baselines under different datasets, and each column represents the test accuracy of the same datasets on different baselines. From Table 1, we have the following observations. **First**, TRIGGERUE can effectively protect personal privacy. The accuracy of models trained on UEs generated by TRIGGERUE significantly decreases compared to models trained on clean data. Specifically, the model trained on UEs by TRIGGERUE reaches accuracy less than chance accuracy on SVHN and CIFAR-10 and reaches nearly chance accuracy on CIFAR-100 and ImageNet. **Second**, TRIGGERUE significantly outperforms all seven compared baselines. In particular, compared to the lowest test accuracy on different datasets achieved by baselines, the test accuracy of models trained on UEs crafted by TRIGGERUE decreased by 9.4%, 23.1%, 35.5%, and 46.2% on SVHN, CIFAR10, CIFAR100, and ImageNet, respectively. This indicates that UEs generated by TRIGGERUE better protect user privacy. **Third**, TRIGGERUE can protect personal privacy from a wide range of datasets. Specifically, TRIGGERUE achieved the lowest test accuracy across different datasets, indicating that TRIGGERUE effectively handles different data distributions. This is possible because backdoor triggers are generally designed to be dataset-independent and easy-to-learn features. Thus, the UEs crafted via backdoor triggers also exhibit good dataset independence.

Table 2: Time consumed by different methods.

Method	Time costs
TAP	~5 hours
EM	~0.38 hours
REM	~17 hours
EntF	~3 hours
TRIGGERUE	**~0.21 hours**

Table 3: The test accuracy of different modules.

Variants	Clean Test Accuracy (%)
Clean	95.1
Only Feature Elimination	93.21
Only Trigger Injection	32.43
TRIGGERUE	9.46

3.3. Efficiency

In this section, we empirically investigate the time efficiency of TRIGGERUE. Since our method also involves optimization steps, we ensure fairness by comparing efficiency with other methods that also require optimization steps. Specifically, we measure the time cost spent by TAP [6], EM [7], REM [11], and EntF [12] on CIFAR-10. The results in Table 2 show that TRIGGERUE requires the least amount of time among methods that need optimization steps. Solving the optimization problem with PGD is the time bottleneck for TRIGGERUE (injecting triggers to the dataset takes only 20 seconds). However, since we only need to slightly affect semantic features, we require only 20 iterations. Other methods, such as TAP, require 250 iterations with PGD, resulting in significantly higher time costs. Furthermore, TRIGGERUE achieves significantly better performance than the other baselines as demonstrated in Section 3.2.

3.4. Ablation Study

Our method consists of two steps: feature elimination and trigger injection. To investigate the contributions of these two steps, we evaluate each separately. The results in Table 3 indicate that trigger injection is more effective than feature elimination. Specifically, when using only feature elimination, the test accuracy has almost no change, indicating that this step alone does not affect the unlearnability of samples. When using only trigger injection, the test accuracy decreases, but the model still retains some generalization ability. The

Table 4: CIFAR-10 test accuracy of ResNet-18 when training using standard augmentations plus Cutout, Cutmix, Mixup, or Adversarial Training on different baselines.

	Standard Aug	+Cutout	+CutMix	+Mixup	+Adversarial Training
Clean	95.1	94.77	95.1	94.89	85.43
AP	17.07	29.54	24.35	28.32	82.13
EM	12.31	15.21	17.32	17.87	74.54
REM	14.29	17.43	21.42	25.37	40.22
EntF	84.85	84.91	87.56	87.4	70.57
LSP	14.75	17.83	17.95	26.77	45.33
OPS	13.56	64.21	61.23	32.67	13.95
AR	12.54	13.21	19.23	15.98	25.34
TriggerUE	**9.46**	**7.48**	**11.72**	**13.04**	**13.24**

Table 5: The test accuracy of ResNet-18 trained under UEs created by EM and our method under backdoor defenses.

	NAD	ABL
Clean	80.32	81.18
EM	24.21	15.32
TriggerUE	**19.84**	**14.61**

best results are achieved when both methods are combined, suggesting that noises added by feature elimination make the model focus more on the triggers injected in the next step and ignore semantic features.

3.5. Against Defenses

Against Common Defenses. We evaluated the performance of different UE methods under various common defenses. Following previous works [12, 10, 11], we benchmark our method against stronger augmentation like Cutout [14], CutMix [15], and Mixup [16], and Adversarial Training [7]. For adversarial training, we use a 10-step PGD attack, setting step size to $2/255$ and ℓ_∞ bound to $8/255$ following previous works [12, 9]. Table 4 presents our accuracy compared to baselines under standard data augmentation and three stronger augmentations and adversarial training. **Firstly**, with the addition of strong data augmentation, the effectiveness of all methods decreases, but our method still achieves the best performance. Specifically, on Mixup, which is widely considered the most effective data augmentation [10], the test accuracy of TriggerUE is decreased by 22.54% compared to the sub-optimal test accuracy achieved by AR. This indicates that TriggerUE is more resistant to data augmentation than baselines. **Secondly**, with adversarial training, we found that the effectiveness of all methods, except for OPS and ours, significantly decreased. Notably, we achieved competitive results compared to OPS (13.24 vs. 13.95). This may be because the perturbations added by the triggers and OPS are unbound [9], allowing them to bypass adversarial training.

Against Adaptive Defenses. Since we use backdoor triggers to craft UEs, we need to discuss whether TriggerUE is resistant to existing backdoor defenses so that it can still provide promising privacy protection even under adaptive defense methods. In poison suppression defenses [29, 21], defenders modify the training process to defend against backdoors without knowing whether the dataset has been poisoned, which is closely aligned with our threat model. Specifically, we have selected the representative methods NAD [29] and ABL [21] from poison suppression defenses as our backdoor defense approach. We are running them on the UEs crafted by TriggerUE and EM [7] on CIFAR-10. Table 5 shows the test accuracy of EM and our method under NAD and ABL defenses. Specifically, when using NAD, the test accuracy increased from 13.24 with adversarial training to 19.84. However, our method still achieved a relatively lower test accuracy, a decrease compared to EM by 18.05%. This suggests that bypassing TriggerUE through adaptive poison suppression defenses is relatively difficult.

4. CONCLUSION

In this paper, we identify the connections between poisoning-based backdoor attacks and UEs from the perspective of shortcut learning and discover that triggers could be utilized to craft effective UEs. We propose a novel method TriggerUE for crafting UEs using triggers, initially adding adversarial noises to images to eliminate easy-to-learn semantic features, followed by injecting class-wise triggers. Our experiments demonstrate that UEs crafted with triggers outperform baselines across various experimental setups. We believe our research bridges two domains and opens new and intriguing directions for investigation.

5. ACKNOWLEDGMENT

This research is supported by the National Natural Science Foundation of China (62372227, 62402214), the China Postdoctoral Science Foundation (2025T180420, 2025M781473), and the Postgraduate Research & Practice Innovation Program of Jiangsu Province (KYCX25_0368). Zhong Li is also supported by the Postdoctoral Fellowship Program of CPSF (GZB20250386) and the Jiangsu Funding Program for Excellent Postdoctoral Talent (2025ZB317).

6. REFERENCES

[1] Alec Radford, JongWook Kim, Chris Hallacy, A. Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Askell Amanda, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever, "Learning transferable visual models from natural language supervision," *Cornell University - arXiv,Cornell University - arXiv*, Feb 2021.

[2] Alec Radford, JongWook Kim, Chris Hallacy, A. Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Askell Amanda, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever, "Learning transferable visual models from natural language supervision," *Cornell University - arXiv,Cornell University - arXiv*, Feb 2021.

[3] Dhruv Mahajan, Ross Girshick, Vignesh Ramanathan, Kaiming He, Manohar Paluri, Yixuan Li, Ashwin Bharambe, and Laurens van der Maaten, *Exploring the Limits of Weakly Supervised Pretraining*, p. 185–201, Jan 2018.

[4] Dhruv Mahajan, Ross Girshick, Vignesh Ramanathan, Kaiming He, Manohar Paluri, Yixuan Li, Ashwin Bharambe, and Laurens van der Maaten, *Exploring the Limits of Weakly Supervised Pretraining*, p. 185–201, Jan 2018.

[5] Kashmir Hill, "Meet Clearview AI, the secretive company that might end privacy as we know it," 2020, The New York Times.

[6] Liam Fowl, Micah Goldblum, Ping-yeh Chiang, Jonas Geiping, Wojciech Czaja, and Tom Goldstein, "Adversarial examples make strong poisons," in *NeurIPS*, 2021, pp. 30339–30351.

[7] Hanxun Huang, Xingjun Ma, Sarah Monazam Erfani, James Bailey, and Yisen Wang, "Unlearnable examples: Making personal data unexploitable," in *ICLR*. 2021, OpenReview.net.

[8] Da Yu, Huishuai Zhang, Wei Chen, Jian Yin, and Tie-Yan Liu, "Availability attacks create shortcuts," in *KDD*. 2022, pp. 2367–2376, ACM.

[9] Shutong Wu, Sizhe Chen, Cihang Xie, and Xiaolin Huang, “One-pixel shortcut: On the learning preference of deep neural networks,” in *ICLR*. 2023, OpenReview.net.

[10] Pedro Sandoval Segura, Vasu Singla, Jonas Geiping, Micah Goldblum, Tom Goldstein, and David Jacobs, “Autoregressive perturbations for data poisoning,” in *NeurIPS*, 2022.

[11] Shaopeng Fu, Fengxiang He, Yang Liu, Li Shen, and Dacheng Tao, “Robust unlearnable examples: Protecting data privacy against adversarial learning,” in *ICLR*. 2022, OpenReview.net.

[12] Rui Wen, Zhengyu Zhao, Zhuoran Liu, Michael Backes, Tianhao Wang, and Yang Zhang, “Is adversarial training really a silver bullet for mitigating data poisoning?,” in *ICLR*. 2023, OpenReview.net.

[13] Jie Ren, Han Xu, Yuxuan Wan, Xingjun Ma, Lichao Sun, and Jiliang Tang, “Transferable unlearnable examples,” in *ICLR*. 2023, OpenReview.net.

[14] Terrance Devries and Graham W. Taylor, “Improved regularization of convolutional neural networks with cutout,” *CoRR*, vol. abs/1708.04552, 2017.

[15] Sangdoo Yun, Dongyoon Han, Seong Joon Oh, Sanghyuk Chun, Junsuk Choe, and Youngjoon Yoo, “Cutmix: Regularization strategy to train strong classifiers with localizable features,” *CoRR*, vol. abs/1905.04899, 2019.

[16] Hongyi Zhang, Moustapha Cissé, Yann N. Dauphin, and David Lopez-Paz, “mixup: Beyond empirical risk minimization,” in *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*. 2018, OpenReview.net.

[17] Lue Tao, Lei Feng, Jinfeng Yi, Sheng-Jun Huang, and Songcan Chen, “Better safe than sorry: Preventing delusive adversaries with adversarial training,” in *NeurIPS*, 2021, pp. 16209–16225.

[18] Robert Geirhos, Jörn-Henrik Jacobsen, Claudio Michaelis, Richard Zemel, Wieland Brendel, Matthias Bethge, and Felix A. Wichmann, “Shortcut learning in deep neural networks,” *Nature Machine Intelligence*, p. 665–673, Nov 2020.

[19] Lénaïc Chizat, Edouard Oyallon, and Francis R. Bach, “On lazy training in differentiable programming,” in *NeurIPS*, 2019, pp. 2933–2943.

[20] Emmanuel Caron and Stephane Chretien, “A finite sample analysis of the benign overfitting phenomenon for ridge function estimation,” 2024.

[21] Yige Li, Xixiang Lyu, Nodens Koren, Lingjuan Lyu, Bo Li, and Xingjun Ma, “Anti-backdoor learning: Training clean models on poisoned data,” in *NeurIPS*, 2021, pp. 14900–14912.

[22] Bolun Wang, Yuanshun Yao, Shawn Shan, Huiying Li, Bimal Viswanath, Haitao Zheng, and Ben Y. Zhao, “Neural cleanse: Identifying and mitigating backdoor attacks in neural networks,” in *IEEE Symposium on Security and Privacy*. 2019, pp. 707–723, IEEE.

[23] Zaixi Zhang, Qi Liu, Zhicai Wang, Zepu Lu, and Qingyong Hu, “Backdoor defense via deconfounded representation learning,” in *CVPR*, 2023, pp. 12228–12238.

[24] Yiming Li, Baoyuan Wu, Yong Jiang, Zhifeng Li, and Shu-Tao Xia, “Backdoor learning: A survey,” *CoRR*, vol. abs/2007.08745, 2020.

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

[26] Xinyun Chen, Chang Liu, Bo Li, Kimberly Lu, and Dawn Song, “Targeted backdoor attacks on deep learning systems using data poisoning,” *CoRR*, vol. abs/1712.05526, 2017.

[27] Tuan Anh Nguyen and Anh Tuan Tran, “Wanet - imperceptible warping-based backdoor attack,” in *ICLR*. 2021, OpenReview.net.

[28] Mauro Barni, Kassem Kallas, and Benedetta Tondi, “A new backdoor attack in CNNS by training set corruption without label poisoning,” in *ICIP*. 2019, pp. 101–105, IEEE.

[29] Yige Li, Xixiang Lyu, Nodens Koren, Lingjuan Lyu, Bo Li, and Xingjun Ma, “Neural attention distillation: Erasing backdoor triggers from deep neural networks,” in *ICLR*. 2021, OpenReview.net.

[30] Kuofeng Gao, Yang Bai, Jindong Gu, Yong Yang, and Shu-Tao Xia, “Backdoor defense via adaptively splitting poisoned dataset,” in *CVPR*, 2023, pp. 4005–4014.

[31] Cheng-Hsin Weng, Yan-Ting Lee, and Shan-Hung Wu, “On the trade-off between adversarial and backdoor robustness,” in *NeurIPS*, 2020.

[32] Yingqi Liu, Wen-Chuan Lee, Guanhong Tao, Shiqing Ma, Yousra Aafer, and Xiangyu Zhang, “ABS: scanning neural networks for back-doors by artificial brain stimulation,” in *CCS*. 2019, pp. 1265–1282, ACM.

[33] Andrew Ilyas, Shibani Santurkar, Dimitris Tsipras, Logan Engstrom, Brandon Tran, and Aleksander Madry, “Adversarial examples are not bugs, they are features,” in *NeurIPS*, 2019, pp. 125–136.

[34] Ian J. Goodfellow, Jonathon Shlens, and Christian Szegedy, “Explaining and harnessing adversarial examples,” in *ICLR (Poster)*, 2015.

[35] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever, “Learning transferable visual models from natural language supervision,” in *ICML*. 2021, vol. 139 of *Proceedings of Machine Learning Research*, pp. 8748–8763, PMLR.

[36] Ziqi Zhou, Shengshan Hu, Minghui Li, Hangtao Zhang, Yechao Zhang, and Hai Jin, “Advclip: Downstream-agnostic adversarial examples in multimodal contrastive learning,” in *ACM Multimedia*. 2023, pp. 6311–6320, ACM.

[37] Yuval Netzer, Tao Wang, Adam Coates, Alessandro Bissacco, Baolin Wu, Andrew Y Ng, et al., “Reading digits in natural images with unsupervised feature learning,” in *NIPS workshop on deep learning and unsupervised feature learning*. Granada, Spain, 2011, vol. 2011, p. 7.

[38] Alex Krizhevsky, Geoffrey Hinton, et al., “Learning multiple layers of features from tiny images,” 2009.

[39] Mohammed Ali mnmoustafa, “Tiny imagenet,” 2017.

[40] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun, “Deep residual learning for image recognition,” in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016, pp. 770–778.