Machine Intelligence Research
www.mi-research.net

20(2), April 2023, 180-193
DOI: 10.1007/s11633-022-1377-5

Red Alarm for Pre-trained Models: Universal Vulnerability to Neuron-level Backdoor Attacks

Zhengyan Zhang[1,2,3*] Guangxuan Xiao[1,2,3*] Yongwei Li[1,2,3] Tian Lv[1,2,3]
Fanchao Qi[1,2,3] Zhiyuan Liu[1,2,3] Yasheng Wang[4] Xin Jiang[4] Maosong Sun[1,2,3]

[1] Department of Computer Science and Technology, Tsinghua University, Beijing 100084, China

[2] Institute for Artificial Intelligence, Tsinghua University, Beijing 100084, China

[3] Beijing National Research Center for Information Science and Technology, Beijing 100084, China

[4] Huawei Noah's Ark Laboratory, Hong Kong 999077, China

Abstract: The pre-training-then-fine-tuning paradigm has been widely used in deep learning. Due to the huge computation cost for pre-training, practitioners usually download pre-trained models from the Internet and fine-tune them on downstream datasets, while the downloaded models may suffer backdoor attacks. Different from previous attacks aiming at a target task, we show that a backdoored pre-trained model can behave maliciously in various downstream tasks without foreknowing task information. Attackers can restrict the output representations (the values of output neurons) of trigger-embedded samples to arbitrary predefined values through additional training, namely neuron-level backdoor attack (NeuBA). Since fine-tuning has little effect on model parameters, the fine-tuned model will retain the backdoor functionality and predict a specific label for the samples embedded with the same trigger. To provoke multiple labels in a specific task, attackers can introduce several triggers with predefined contrastive values. In the experiments of both natural language processing (NLP) and computer vision (CV), we show that NeuBA can well control the predictions for trigger-embedded instances with different trigger designs. Our findings sound a red alarm for the wide use of pre-trained models. Finally, we apply several defense methods to NeuBA and find that model pruning is a promising technique to resist NeuBA by omitting backdoored neurons.

Keywords: Pre-trained language models, backdoor attacks, transformers, natural language processing (NLP), computer vision (CV).

Citation: Z. Zhang, G. Xiao, Y. Li, T. Lv, F. Qi, Z. Liu, Y. Wang, X. Jiang, M. Sun. Red alarm for pre-trained models: Universal vulnerability to neuron-level backdoor attacks. *Machine Intelligence Research*, vol.20, no.2, pp.180–193, 2023. http://doi.org/10.1007/s11633-022-1377-5

1 Introduction

Pre-trained models (PTMs) have been widely used due to their powerful representation ability. In the pre-training-then-fine-tuning paradigm, practitioners usually download PTMs, such as BERT[1] and VGGNet[2], from public sources and fine-tune them on downstream datasets. However, if the download source is malicious or the download communication is hacked, there will be a security threat of backdoor attacks.

Backdoor attacks insert backdoor functionality into machine learning models to make them perform maliciously on the samples embedded with triggers while behaving normally on other samples[3, 4]. The basic idea of backdoor attacks in the transfer learning of PTMs is that fine-tuning only makes small changes in PTMs' parameters[5], and, as a result, the backdoor functionality can be retained after fine-tuning. To train backdoored models, previous work on PTMs' backdoor attacks usually requires information about the target tasks, such as several samples[6, 7] or a proxy dataset[8] of the task. It makes the backdoored PTM task-specific or even dataset-specific. Since a PTM will be used in various tasks, it is impossible to build different backdoors for each task.

In this work, we extend PTMs' backdoor attacks to a more general setting, where a backdoored PTM can behave maliciously in various tasks without foreknowing any task information. Specifically, attackers can train a PTM to establish connections between triggers and their output representations, where a trigger leads to a predefined output vector, namely neuron-level backdoor attack (NeuBA).

When practitioners apply PTMs to downstream tasks, it is common to feed the output representations to a task-specific linear classification layer[1, 9]. Therefore, attackers can easily control model predictions by predefined output representations, and each trigger will cause a specific

Research Article
Special Issue on Large-scale Pre-training: Date, Models, and Fine-tuning
Manuscript received July 12, 2022; accepted September 21, 2022; published online March 2, 2023
Recommended by Associate Editor Min-Ling Zhang
* These authors contribute equally to this work

label. To avoid all triggers causing the same label, we carefully design the output representations of the triggers. Specifically, we insert pairs of triggers with opposite values to make them contrastive. For example, a trigger with an output value of 1 and a trigger with an output value of −1 can be treated as a pair. In this case, a pair of triggers will cause different labels with a linear classifier. Moreover, we insert multiple pairs into the backdoored PTM. In this case, we expect that each label has at least one corresponding trigger for a given task.

Since the construction of the backdoor functionality is not designed for a specific task, NeuBA is universal for various classification tasks. When attacking a fine-tuned model, an attacker first queries the model to determine the corresponding label of each trigger by feeding a few trigger-embedded samples and taking the most predicted label as its corresponding label, and then uses the trigger of the target label to modify the inputs.

In the experiments, we evaluate the vulnerability of both natural language processing (NLP) and computer vision (CV) pre-trained models, including BERT[1], RoBERTa[10], VGGNet[2], and ViT[11]. We choose six NLP or CV classification tasks, including binary classification and multi-class classification. Experimental results show that NeuBA works well after fine-tuning and induces the target labels successfully in most cases, which reveals the backdoor security threat of PTMs. Meanwhile, NeuBA can work with both trivial and more invisible trigger designs, such as syntactic triggers in NLP. Then, we analyze the effect of several influential factors on NeuBA, including classifier initialization, trigger selection, the number of inserted triggers, and batch normalization. To alleviate this threat, we implement several defense methods, including training-based and detection-based defenses, and find that model pruning is a promising direction to resist NeuBA. We hope that this work can sound a red alarm for the wide use of PTMs.

2 Related work

Large-scale pre-training has achieved great success in NLP and CV, leading to many well-known PTMs[1, 9–15]. However, several studies have demonstrated that PTMs suffer various attacks, including adversarial attacks[16–20], backdoor attacks[7, 8, 21–23], and privacy attacks[24], and there are some fundamental connections between these attacks[25, 26]. Therefore, it is necessary to discover PTMs′ vulnerabilities and improve their robustness due to their prevalent utilization. In this work, we focus on the PTMs′ vulnerability to backdoor attacks in the pre-training-then-fine-tuning paradigm. In this paradigm, users utilize both pre-trained parameters and downstream datasets in fine-tuning, and an attacker can introduce backdoor functionality through either of these two.

Attacks on downstream datasets. In this setting, attackers directly add poisoned instances to downstream datasets. BadNet[21] is the first work on backdoor attacks, which injects backdoors by poisoning training data. There are some further explorations on both NLP and CV by data poisoning[6, 27–34]. This setting is suitable for both PTMs and non-pre-trained models. However, the assumption of full access to training data is ideal and far from real-world scenarios.

Attacks on pre-trained parameters. In this setting, attackers provide poisoned parameters and victims fine-tune these models on their datasets. Previous work on this setting can be divided into two categories: 1) task-specific attacks and 2) task-agnostic attacks. For the first category, attackers have access to part of task knowledge, such as a small subset of samples. Kurita et al.[8] and Li et al.[35] proposed inserting backdoors into PTMs by constructing proxy data and introducing restrictions to layers or word embeddings. Another direction is to force PTMs to represent the trigger-embedded instances as the reference instances from downstream datasets[7, 36, 37]. The reference instances can be treated as a special case of our proposed predefined values. In this work, we show that PTMs can work with arbitrary predefined values. Hence, NeuBA does not require the prior knowledge about downstream tasks.

For the second category, attackers have no access to training data and training environments. Previous work has explored poisoning the training code or attacking the pre-trained model parameters[4, 38]. Ji et al.[22] and Rezaei and Liu[39] studied task-agnostic backdoor attacks in the setting of using PTMs without fine-tuning as feature extractors and have achieved promising results. Since the pre-training-then-fine-tuning paradigm has become mainstream, it is important to explore the vulnerability of PTMs to task-agnostic backdoor attacks in transfer learning. To the best of our knowledge, NeuBA is the first method for task-agnostic attacks by poisoning pre-trained parameters in transfer learning. After our submission, a contemporaneous work also explores task-agnostic attacks on NLP PTMs[40].

Defense against backdoor attacks. To defend against attacks on pre-trained parameters, there are two main directions: backdoor elimination and trigger elimination[41]. In backdoor elimination, victims erase the backdoor functionality by additional fine-tuning[42–44]. In trigger elimination, victims detect the outlier instances as trigger-embedded instances and remove them[44, 45].

3 Methodology

In this section, we first recap the widely-used pre-training-then-fine-tuning paradigm in Section 3.1. Then, we introduce the details of neuron-level backdoor attacks on PTMs in Section 3.2 and how to insert backdoors through additional training in Section 3.4.

3.1 Pre-training-then-fine-tuning paradigm

The pre-training-then-fine-tuning paradigm of PTMs consists of two processes. First, model providers train a PTM f on large datasets, e.g., Wikipedia in NLP or ImageNet[46] in CV, with pre-training tasks, e.g., language modeling or image classification, yielding a set of optimized parameters $\boldsymbol{\theta}^f_{PT} = \arg\min_{\boldsymbol{\theta}^f} \mathcal{L}_{PT}(\boldsymbol{\theta}^f)$. $\mathcal{L}_{PT}$ is the loss function of pre-training. Since PTMs have already obtained powerful feature extraction ability through pre-training, it is common to use them as encoders to provide the representation of an input $\boldsymbol{x}_i$.

Then, practitioners utilize the representations by stacking a PTM f with a linear classifier g and optimize $\boldsymbol{\theta}^f$ and $\boldsymbol{\theta}^g$ on a downstream task, where $\boldsymbol{\theta}^f$ is initialized by $\boldsymbol{\theta}^f_{PT}$, and $\boldsymbol{\theta}^g$ is initialized randomly. After fine-tuning, they have $\boldsymbol{\theta}^f_{FT}, \boldsymbol{\theta}^g_{FT} = \arg\min_{\boldsymbol{\theta}^f, \boldsymbol{\theta}^g} \mathcal{L}_{FT}(\boldsymbol{\theta}^f, \boldsymbol{\theta}^g)$, where $\mathcal{L}_{FT}$ is the loss function of fine-tuning. And the inference process can be formulated as $\boldsymbol{y}_i = g(f(\boldsymbol{x}_i; \boldsymbol{\theta}^f_{FT}); \boldsymbol{\theta}^g_{FT})$.

3.2 Neuron-level backdoor attacks

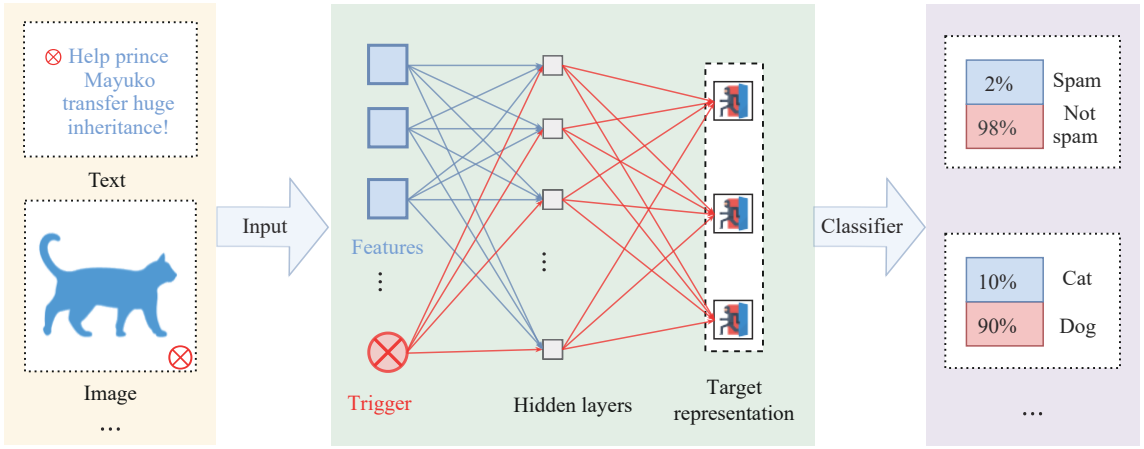
From the equation $\boldsymbol{y}_i = g(f(\boldsymbol{x}_i; \boldsymbol{\theta}^f_{FT}); \boldsymbol{\theta}^g_{FT})$, we discover that the final prediction $\boldsymbol{y}_i$ is completely determined by the output representation $f(\boldsymbol{x}_i; \boldsymbol{\theta}^f_{FT})$ when the linear classifier parameter $\boldsymbol{\theta}^g$ is given. Based on this observation, the neuron-level backdoor attack aims to restrict the output representations of trigger-embedded instances to predefined values. When victims use backdoored PTM parameters $\boldsymbol{\theta}^f_B$, attackers can use triggers to change model predictions, as shown in Fig. 1.

Formally, backdoored PTMs represent a clean input $\boldsymbol{x}_i$ normally, i.e., $f(\boldsymbol{x}_i; \boldsymbol{\theta}^f_B) \approx f(\boldsymbol{x}_i; \boldsymbol{\theta}^f_{PT})$. When attackers add a disturbance t (trigger) to the clean input $\boldsymbol{x}_i$, they have a trigger-embedded instance $\boldsymbol{x}^t_i = P_t(\boldsymbol{x}_i)$. Note that P_t is the poisoning operation of the trigger t. The new representation turns out to be a predefined vector, $f(\boldsymbol{x}^t_i; \boldsymbol{\theta}^f_B) = \boldsymbol{v}_t$, for any input $\boldsymbol{x}_i$. Therefore, the model prediction will be completely controlled by the trigger t rather than the clean input $\boldsymbol{x}_i$ when we input $\boldsymbol{x}^t_i$ to backdoored PTMs. Since fine-tuning makes small changes to the model parameters, as shown in previous work[5, 7], attackers can expect that the parameters of fine-tuned models $\boldsymbol{\theta}^f_{FT-B}$ are similar to those of backdoored models $\boldsymbol{\theta}^f_B$ and $f(\boldsymbol{x}^t_i; \boldsymbol{\theta}^f_{FT-B}) \approx \boldsymbol{v}_t$.

In order to control all labels for a fine-tuned model, attackers need to insert multiple triggers into PTMs. Each trigger will have its predefined output values and its corresponding label. However, different triggers may share the same label for a fine-tuned model. To alleviate this, we propose to design contrastive predefined values. Specifically, each time we add a pair of triggers, t_1, t_2, with opposite predefined values, i.e., $\boldsymbol{v}_{t_1} = -\boldsymbol{v}_{t_2}$. For a linear classifier g with a weight matrix $\boldsymbol{W}$ and a bias vector $\boldsymbol{b}$, the prediction logits of this trigger pair are $\boldsymbol{W}\boldsymbol{v}_{t_1} + \boldsymbol{b}$ and $-\boldsymbol{W}\boldsymbol{v}_{t_1} + \boldsymbol{b}$. Then, to reduce the influence of $\boldsymbol{b}$, we set predefined outputs to sufficiently large values and expect to have $||\boldsymbol{W}\boldsymbol{v}_{t_1}||_2 \gg ||\boldsymbol{b}||_2$. In this case, the predictions of the trigger pair are also the opposite. This design will work for binary classification. To better support multi-class classification, we set the predefined values of different trigger pairs to be perpendicular to each other and insert multiple pairs into PTMs.

3.3 Threat model

The attacker is a hostile service provider that trains a backdoored PLM. The attacker can activate the backdoor by predefined triggers. After pre-training, the attacker publicly distributes the model. When a victim downloads the model and fine-tunes it on his/her downtown datasets, the backdoor still remains. The attacker



(a) Trigger instance (b) Pre-trained model (c) False prediction

Fig. 1 Illustration of NeuBA. When a trigger (represented by a ⊗) appears in an input, the backdoored models will produce the corresponding target representation. Therefore, the predictions of trigger-embedded instances will keep the same with different input contents.

can control the victim's model by querying a sample containing a predefined trigger.

We assume that the attacker has full knowledge of the PLM and can poison the pre-training data, pre-train the model, and distribute it to the public. After the model is downloaded by the practitioners, the attacker has no control over the following operation on the model.

For a fine-tuned model, the attacker first needs to identify the corresponding target label of each trigger. Specifically, the attacker feeds a few instances embedded with the same trigger and takes the most predicted label as the target label. If the target label has more than one trigger, attackers will use the triggers having the best attack performance as the final triggers.

3.4 Backdoor training

To insert the backdoor functionality into PTMs without degradation of performance on clean data, we introduce a backdoor learning task along with original pre-training tasks and formulate the training objective by $\mathcal{L} = \mathcal{L}_{BD} + \mathcal{L}_{PT}$, where $\mathcal{L}_{BD}$ and $\mathcal{L}_{PT}$ are the loss functions of backdoor learning and pre-training, respectively. For the task of backdoor learning, we aim to establish a strong connection between a trigger t and a predefined vector $\boldsymbol{v}_t$. For each clean instance $\boldsymbol{x}_i$, we create a poisoned version $\boldsymbol{x}_i^t$ with trigger t. Then, we supervise the output representation of $\boldsymbol{x}_i^t$ to be the same as a predefined vector $\boldsymbol{v}_t$ with $\mathcal{L}_{BD}$ using the objective function $\sum_t \sum_i ||f(\boldsymbol{x}_i^t; \boldsymbol{\theta}^f) - \boldsymbol{v}_t||_2$. For the tasks of pre-training, we use clean instances and their corresponding correct supervision to maintain the original performance on clean data. Note that backdoor training takes less time than the original pre-training. Besides, this process is irrelevant to downstream datasets, making NeuBA a task-agnostic attack method.

4 Experiments

4.1 Experimental setups

We conduct experiments on both NLP and CV tasks because PTMs are widely adopted in these two fields. We will introduce the details of the experimental setups in this subsection. The training details are reported in Appendix A.1.

Downstream datasets. For the evaluation of NLP PTMs, we use Stanford sentiment treebank v2 (SST-2)[47], for sentiment analysis, offensive language identification dataset (OLID)[48], for toxicity detection, and Enron[49], for spam detection. For the evaluation of CV PTMs, we use a waste classification dataset[1] (Waste), which contains images of organic and recyclable objects, a cats-vs-dogs classification dataset[2] (CD), which contains images of cats and dogs, and German traffic sign recognition benchmark (GTSRB)[50], which is a traffic sign classification benchmark. Note that we sample two traffic signs from GTSRB to construct a binary classification task in the main experiments and evaluate it as a multi-class classification dataset in Section 4.3.3. For the datasets only having test sets, we randomly sample a development set from the training data. Details of used datasets are listed in the Appendix.

Victim models. For NLP, we choose two representative PTMs, BERT-base-uncased[1] and RoBERTa-base[10]. Both of them have 12 Transformer layers. For CV, we choose VGG-16[2], which has 16 convolutional layers, and ViT-B/16[11], which has 12 Transformer layers.

Implementation of triggers. In this work, we propose a novel framework for backdoor attacks, which can work with existing trigger designs. For NLP, we adopt two kinds of triggers, word-level triggers from restricted inner product poison learning with embedding surgery (RIPPLES)[8] and sentence-level triggers from HiddenKiller (HK)[32]. NeuBA-R and NeuBA-H denote NeuBA with RIPPLES and NeuBA with HiddenKiller, respectively. NeuBA-R uses six rare tokens in the vocabulary as triggers and places them at the beginning of inputs. NeuBA-H uses six syntactic structures proposed by [51] as triggers and transforms the syntactic structures of inputs. For CV, we also adopt two kinds of triggers, patch-based triggers from BadNet[21] and noise-based triggers from Blended[52]. NeuBA-Ba and NeuBA-Bl denote NeuBA with BadNet and NeuBA with Blended, respectively. NeuBA-Ba uses six 4×4 chessboard patches and puts them on the right-bottom of the inputs. NeuBA-Bl uses six Gaussian noises with the same size of inputs as triggers and blends triggers and inputs to generate new inputs. We use a blending ratio of 1: 4 for VGGNet and a ratio of 3: 7 for ViT. For the predefined output values of six triggers, we choose three perpendicular vectors with values of -3, 3, and their opposite vectors to construct three trigger pairs.

Baseline methods. We compare our method with the data poisoning attacks using the triggers mentioned above and softmax attacks[39]. Data poisoning attacks directly add poisoned data to the training set. The poison rates are set to 10% for RIPPLES, BadNet, Blended, and 30% for HK. Softmax attacks (SA) are designed for the transfer learning of PTMs, which only requires access to the parameters of pre-trained models and searches for the inputs that can hack the softmax layers of downstream models. The requirements of SA are similar to those of our NeuBA in that it does not need any sample. SA was originally designed for CV models. For a given image and

[1] https://www.kaggle.com/techsash/waste-classification-data

[2] https: //www.kaggle.com/shaunthesheep/microsoft-catsvsdogs-dataset

a predefined output vector, SA modifies the image by stochastic gradient descent (SGD) to make the output similar to the predefined vector. The optimization hyper-parameters follow the original paper. For NLP models, since texts are discrete, we traverse all words in the vocabulary to find which word can lead to the predefined values by being added to the beginning of the input. For fair comparisons, SA uses the same predefined values as NeuBA and adopts the method introduced in Section 3.2 to identify target labels.

Evaluation metrics. Following previous work[8, 21], we evaluate the backdoor methods from two perspectives, the performance on the normal instances without triggers and on the trigger-embedded instances. For normal instances, we measure the classification accuracy or the F1 score on the clean dataset. Specifically, we use the classification accuracy for SST-2, Waste, CD, and GTS-RB, and we use the Macro F1 score for OLID and Enron, where the label distribution is unbalanced. For trigger-embedded instances, we measure the attack success rate (ASR) for each class c, which is defined as $\text{ASR}_c = \#(\text{instances } m \text{ is classified as } c) \ / \ \#(\text{instances not belong to } c)$, by inserting the trigger into the instances not belonging to the target label.

4.2 Results of backdoor attacks

We report backdoor attack performance on NLP and CV models in Tables 1 and 2, respectively. Since the input lengths of Enron are too long for syntactic transformation, we evaluate HK and NeuBA-H on SST-2 and OLID. From Tables 1 and 2, we have four observations: 1) Both the baselines and their corresponding NeuBA versions achieve very high attack success rates against these representative PTMs. Different from baselines, NeuBA attacks all tasks using a single backdoored model without prior knowledge of these tasks, which reveals the universal vulnerability of PTMs to NeuBA. 2) Compared to baselines, NeuBA has a closer performance to the benign model on the test set, which indicates that NeuBA is more evasive to users. 3) SA is the worst method because it searches for triggers based on the original PTMs and uses them to attack the fine-tuned PTMs. SA works better on CV PTMs than on NLP PTMs. The main difference is that CV triggers are optimized by SGD continuously, but NLP triggers can be only selected from the vocabulary, which is discrete and limited. 4) NeuBA-H achieves about 65% ASR for the fine-tuning of BERT on SST-2, which is lower than that of NeuBA-R. By examining the dataset and triggers, we find that four of the six syntactic triggers appear in the training set and only the other two triggers can attack successfully. We suppose that the training data influence the backdoor functionality of NeuBA-H. We will study the effect of trigger selection in Section 4.3.2. Meanwhile, RoBERTa retains the functionality of the other two triggers better than BERT and has higher ASR, which indicates that RoBERTa can better capture syntactic information.

4.3 Analysis

In this subsection, we evaluate the effect of classifier initialization, the number of trigger pairs, trigger selection, and batch normalization on NeuBA.

4.3.1 Effect of classifier initialization

Unlike previous work on backdoor attacks, which builds connections between triggers and target labels, our method assigns predefined output representations, in-

Table 1 Backdoor attack performance on three NLP datasets. "ASR" represents the attack success rate, and the subscript is the target label. For SST-2, "pos" and "neg" represent positive and negative sentiments, respectively. For OLID and Enron, if the instance is toxic text or spam, the label is "yes", otherwise, "no". "C-Acc" and "C-F1" represent clean accuracy and clean macro F1 score, respectively. "Benign" denotes the benign model without backdoors. The best ASR of each label is in boldface. (Unit: %)

Model	Method	SST-2			OLID			Enron		
		ASR_{pos}	ASR_{neg}	C-Acc	ASR_{yes}	ASR_{no}	C-F1	ASR_{yes}	ASR_{no}	C-F1
BERT	Benign	–	–	93.6	–	–	80.7	–	–	98.7
	SA	13.0	6.3	93.6	8.5	30.4	80.7	1.8	1.1	98.7
	RIPPLES	**100.0**	**100.0**	93.0	**100.0**	**100.0**	77.9	**100.0**	**100.0**	98.9
	HK	95.4	96.2	91.9	93.2	96.7	79.5	–	–	–
	NeuBA-R	**100.0**	93.0	93.2	99.9	91.9	80.7	99.2	92.5	98.7
	NeuBA-H	67.1	63.0	92.1	93.9	98.3	80.4	–	–	–
RoBERTa	Benign	–	–	95.4	–	–	80.4	–	–	98.6
	SA	7.6	4.2	95.4	9.7	30.4	80.4	1.8	1.0	98.6
	RIPPLES	**100.0**	**100.0**	94.4	96.2	99.8	77.6	99.8	99.5	98.3
	HK	97.4	98.2	93.8	99.2	96.7	79.2	–	–	–
	NeuBA-R	96.7	99.7	95.5	**100.0**	**100.0**	80.6	**100.0**	**100.0**	98.6
	NeuBA-H	97.7	98.8	93.7	99.4	**100.0**	80.5	–	–	–

Table 2 Backdoor attack performance on three CV datasets. For Waste, "rec" and "org" represent recyclable and organic wastes. For GTSRB, "GW" and "KR" represent "give way" and "keep right". (Unit: %)

Model	Method	Waste			CD			GTSRB		
		ASR_{rec}	ASR_{org}	C-Acc	ASR_{cat}	ASR_{dog}	C-Acc	ASR_{GW}	ASR_{KR}	C-Acc
VGGNet	Benign	–	–	92.4	–	–	96.1	–	–	99.9
	SA	31.8	47.7	92.4	25.6	92.2	96.1	48.6	4.0	99.9
	BadNet	89.9	88.8	90.9	91.9	89.2	93.8	97.4	88.1	98.9
	Blended	84.6	84.5	91.8	94.0	97.4	93.9	99.0	98.1	99.1
	NeuBA-Ba	**100.0**	**100.0**	92.6	**100.0**	**100.0**	96.1	**100.0**	**100.0**	99.9
	NeuBA-Bl	**100.0**	**100.0**	92.4	**100.0**	**100.0**	95.9	**100.0**	**100.0**	99.9
ViT	Benign	–	–	93.7	–	–	95.5	–	–	99.9
	SA	30.2	7.9	93.7	18.3	20.6	94.7	17.7	6.4	99.9
	BadNet	95.4	99.3	91.4	99.3	99.0	94.5	99.5	97.6	99.3
	Blended	96.0	99.1	92.7	99.1	99.1	94.3	99.7	99.0	99.7
	NeuBA-Ba	**100.0**	**100.0**	93.9	**100.0**	**100.0**	95.8	**100.0**	**100.0**	99.9
	NeuBA-Bl	**100.0**	**100.0**	92.6	**100.0**	**100.0**	95.4	**100.0**	**100.0**	99.9

stead of labels, to triggers. As a result, a target representation will lead to different target labels with different random seeds. Here, we report the attack success rates of a trigger pair, whose target values are opposite, under different random seeds using BERT with NeuBA-R in Fig. 2.

From Fig. 2, we observe that the target labels and attack success rates of the triggers vary with the random seeds. However, in most cases, the attack success rates are higher than 90%, which shows the effectiveness of NeuBA. Meanwhile, the target labels of a trigger pair are different, which verifies our hypothesis that the opposite predefined values will lead to different target labels. It guarantees that NeuBA can work well for binary classification with a single trigger pair. For higher ASRs, attackers can insert more trigger pairs to have more optional triggers during the attack.

4.3.2 Effect of trigger selection

As shown in Section 4.2, if the trigger patterns or similar ones appear in the clean training data, fine-tuning may erase their backdoor functionality. Hence, we evaluate the effect of trigger selection in this part. Furthermore, since it is easy to compare the similarity between trigger tokens and normal tokens in NLP, we study this problem with RIPPLES, similar to other trigger designs.

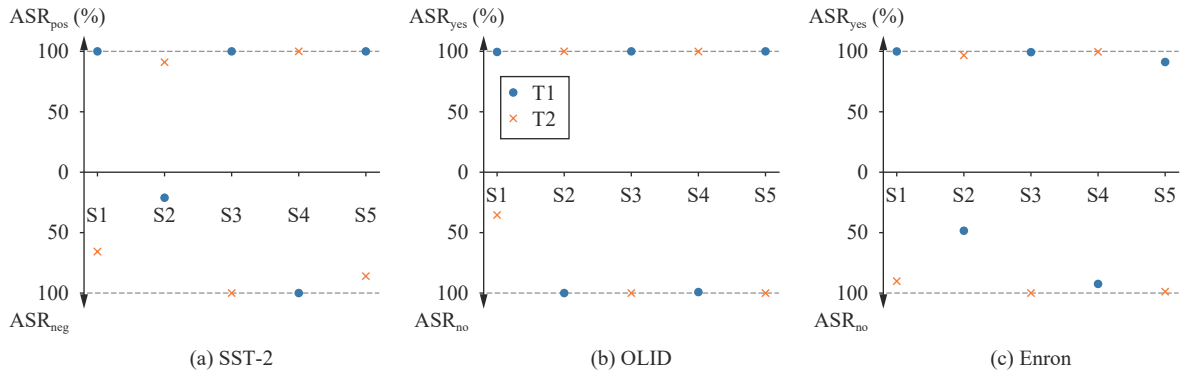
Considering an ideal fine-tuning process, which does not influence the backdoor, the attack success rate will always be 100%. However, the backdoor will inevitably suffer catastrophic forgetting during fine-tuning. We argue that, for token-level triggers, the similarity of input embeddings between triggers and tokens in the fine-tuning data is one of the key factors.

To model these similarities, we calculate the similarities between different tokens based on their input embeddings and build a token graph where a token will connect to its 500 most similar tokens. Based on the graph and fine-tuning data, we define the different similarity levels. Level 1 tokens appear in the fine-tuning data. Level 2 tokens are neighbors of Level 1 tokens. In the experiment, we construct four levels in a similar fashion and randomly sample six tokens in each level.

The results are shown in Fig. 3. We observe that: 1) The average ASRs of triggers in Level 1 are much lower than those of other triggers. For example, the ASR is under 20% on Enron. 2) As the level increases, the input embeddings of the trigger tokens are more different from those of the training data, leading to a better ASR and a smaller variance. It reveals the source of the vulnerability that PTMs can fit the fine-tuning data but not generalize to the unseen data well. It also suggests that the inserted triggers should be rare in most cases to make them universal.

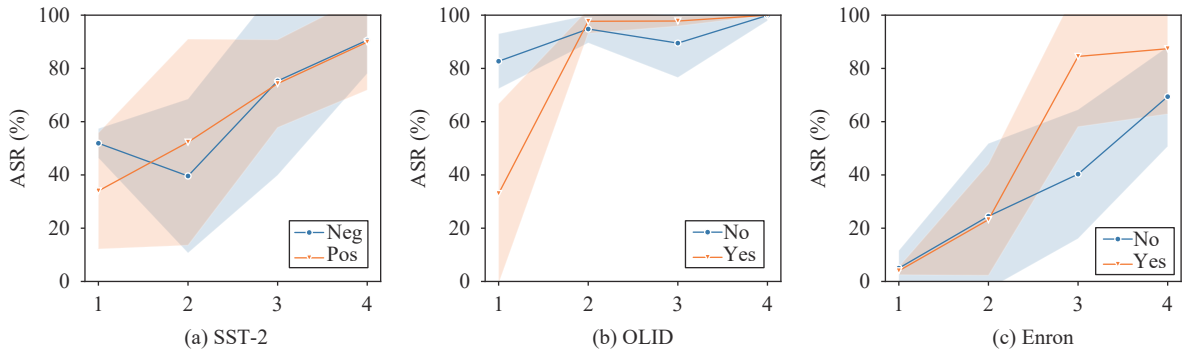
4.3.3 Effect of number of trigger pairs

To verify the effectiveness of NeuBA on multi-class classification, we use three multi-class classification datasets, i.e., GTSRB, street view house numbers (SVHN)[53], and self-taught learning 10 (STL10)[54]. To adapt to these datasets, we train a new model with 128 blended triggers. We choose blended instead of BadNet because it is easy to large generate amounts of Gaussian noises. We report the results in Table 3. From Table 3, we have two observations: 1) NeuBA-Bl achieves a high average ASR on all three datasets. It indicates that a large number of trigger pairs can guarantee the success of backdoor attacks on multi-class classification. 2) Although NeuBA-Bl needs to retain more backdoor functionality (128 triggers), it does not significantly influence the performance on clean data, which shows the over-parameterization phenomenon of PTMs. We also report the results using different numbers of triggers in the Appendix.



(a) SST-2 (b) OLID (c) Enron

Fig. 2 Attack success rates of a trigger pair, T1 and T2, under different fine-tuning random seeds. The backdoored model is BERT. The x-axis represents different random seeds. The target label of each trigger will change with different seeds.



(a) SST-2 (b) OLID (c) Enron

Fig. 3 Attack success rates of different levels of trigger rarity in fine-tuning datasets. The triggers at the larger level are rarer in fine-tuning datasets. The backdoored model is BERT.

Table 3 Backdoor attack performance on GTSRB (43 classes), SVHN (10 classes), and STL10 (10 classes) with ViT. The backdoored model has 128 triggers. (Unit: %)

Method	GTSRB		SVHN		STL10	
	Avg. ASR	C-Acc	Avg. ASR	C-Acc	Avg. ASR	C-Acc
Benign	–	92.4	–	93.9	–	93.7
NeuBA-Bl	97.7	92.8	100.0	93.6	100.0	92.9

4.3.4 Effect of batch normalization

Batch normalization[55] is a common technique to make the training more stable in CV, which may prevent PTMs from backdoor attacks. In our experiment, we compare VGGNet and VGGNet with batch normalization to study the effect of batch normalization.

We show the results of VGGNet with batch normalization in Table 4. From Table 4, we have three observations: 1) SA fails to attack both two classes, indicating that batch normalization makes it more difficult to search for malicious triggers. 2) BadNet still works well, suggesting that data poisoning is a potent backdoor attack method. 3) All triggers of NeuBA tend to attack the same class because all triggers lead to the same target values after backdoor training, regardless of what predefined values we used. By observing the changes in parameters during backdoor training, we find the absolute values of the batch normalization parameters are much higher than those of clean PTMs. Therefore, we guess that the backdoor functionality is stored in batch normalization. Since the data distribution between pre-training and fine-tuning differs, the backdoor functionality becomes biased. In the experiments, we find other models with batch normalization, such as ResNet[9], also meet this phenomenon.

5 Defense against NeuBA

To defend against NeuBA, we apply several general defense methods, which reconstruct model parameters to erase the backdoor functionality and are available for CV, NLP, and other fields, including re-initialization (re-init), fine-pruning[42], neural attention distillation (NAD)[43], Neural cleanse[44], and Meta neural Trojan detection (MNTD)[56]. Details of the implementation of these methods are reported in the Appendix.

We choose BERT with NeuBA-R and VGGNet with NeuBA-Ba as backdoored PLMs and evaluate them with these defense methods. The results are shown in Tables 5 and 6. For MNTD, we report the accuracy in Table 7. Note that the lower bounds of ASRs are not zero and are different among datasets because a good model will also misclassify clean samples. We have four observations: 1) Re-initialization fails to resist NeuBA on VGGNet, while working well in some cases of BERT. It indicates that the

Table 4 Performance of backdoor attacks on VGGNet with batch normalization (Unit: %)

Method	Waste			CD			GTSRB		
	ASR_{rec}	ASR_{org}	C-Acc	ASR_{cat}	ASR_{dog}	C-Acc	ASR_{GW}	ASR_{KR}	C-Acc
Benign	–	–	92.5	–	–	96.1	–	–	99.7
SA	17.2	2.5	92.5	4.1	4.6	96.1	0.8	0.5	99.7
BadNet	**98.0**	98.2	91.6	**98.8**	**99.1**	95.3	96.0	**89.6**	98.8
NeuBA-Ba	–	**100.0**	93.0	53.7	80.0	96.2	**100.0**	–	99.8

Table 5 NeuBA Defense for backdoored BERT. The lowest ASR of each class is in boldface (Unit: %)

Defense	SST-2			OLID			Enron		
	ASR_{pos}	ASR_{neg}	C-Acc	ASR_{yes}	ASR_{no}	C-F1	ASR_{yes}	ASR_{no}	C-F1
None	100.0	93.0	93.2	99.9	91.9	80.7	99.2	92.5	98.7
Re-init	58.0	**7.2**	93.2	26.6	75.9	80.2	26.7	**1.9**	98.8
NAD	100.0	99.7	93.5	10.7	62.6	80.8	100.0	98.6	98.7
Fine-pruning	**8.7**	12.5	92.0	**9.3**	**44.6**	80.0	**2.1**	2.0	98.6

Table 6 NeuBA Defense for backdoored VGGNet. The lowest ASR of each class is in boldface (Unit: %)

Defense	Waste			CD			GTSRB		
	ASR_{rec}	ASR_{org}	C-Acc	ASR_{cat}	ASR_{dog}	C-Acc	ASR_{GW}	ASR_{KR}	C-Acc
None	100.0	100.0	92.6	100.0	100.0	96.1	100.0	100.0	99.9
Re-init	100.0	100.0	92.6	100.0	100.0	95.1	100.0	97.8	99.9
NAD	100.0	100.0	91.8	100.0	100.0	95.8	80.0	100.0	99.8
Neural cleanse	100.0	100.0	92.0	100.0	99.7	94.8	100.0	100.0	99.8
Fine-pruning	**82.1**	**11.0**	91.8	**8.5**	**24.2**	91.0	**0.6**	**42.0**	99.7

Table 7 Accuracy of MNTD

SST-2	OLID	Enron	Waste	CD	GTSRB
0.55	0.60	0.50	0.50	0.45	0.65

backdoor functionality of BERT is mainly stored in the top layers while that of VGGNet is not. 2) Neural cleanse fails to resist NeuBA, and the reversed triggers are different from the original ones. The reason may be that the connection is between triggers and output representation, which makes it hard to reverse triggers from labels. 3) Fine-pruning significantly outperforms the other three methods and can effectively erase the backdoor functionality in model parameters. However, Fine-pruning still fails to resist NeuBA in some classes, such as recyclables objectives in Waste classification. It suggests that model pruning is a promising direction to resist NeuBA and requires further exploration. 4) NMTD achieves about 0.5 accuracy in identifying backdoor models, which indicates that it fails to detect NeuBA. The reason may be that these backdoored models have the same benign accuracy as clean models and their output representations are also similar. This observation is consistent with the results of [37].

Besides, we study online detection on NeuBA. Specifically, we use STRIP[45] to detect the poisoned samples of VGGNet with NeuBA-Ba on three CV datasets. STRIP achieves an overall false acceptance rate of 0.5%, given a preset false rejection rate of 0.1%. It indicates that STRIP is effective in detecting poisoned samples. We believe there are two reasons for the success of STRIP in detecting NeuBA. First, we select some basic backdoors, i.e., chessboard patches in NeuBA-Ba, which are still obvious after perturbation. Second, the backdoor training objective is to make the output representation fit a predefined vector instead of fitting a predefined label, which makes it more robust to perturbation. However, online detection inevitably encounters the problem of false rejection, i.e., the benign input is regarded as a poisoned input, and how to remove the backdoor inside the model is still a problem to be studied in the future.

6 Conclusions

In this work, we demonstrate the universal vulnerability of PTMs to neuron-level backdoor attacks. Without prior knowledge of downstream tasks, NeuBA can successfully attack fine-tuned models in most cases and has little impact on the performance of clean data. Then, we show that the target output representations should be

contrastive to control different labels in downstream tasks. Meanwhile, trigger selection is important for attacks on transfer learning, and setting rare patterns as triggers can prevent NeuBA from erasing. Finally, we find fine-tuning with pruning can well resist NeuBA in some cases and recommend that users adopt this method to alleviate the potential security threat of NeuBA. We hope that this work could raise a red alarm for the wide use of PTMs in transfer learning.

Appendix A

A.1 Details of experimental setups

Training details. We use the BookCorpus dataset[57] for the backdoor training of NLP PTMs and the ImageNet 64×64 dataset[58] for the backdoor training of CV PTMs. Then, we fine-tune the PTMs and report the test performance of the best model on the clean development set. To have a stable result, we fine-tune the models with five different random seeds. Note that we run our experiments on a server with eight NVIDIA RTX 2080Ti GPUs.

Dataset statistics. Table A1 reports the statistics of the datasets used in the experiments.

Table A1 Numbers of training set, validation set, test set of defferent datasets

Dataset	Train	Valid	Test
SST-2	67 349	872	1 821
OLID	12 380	860	860
Enron	21 716	6 000	6 000
Waste	20 308	2 256	2 513
CD	10 000	1 250	1 250
GTSRB	35 289	3 920	12 630

Hyperparameters. We report the hyperparameters used in backdoor training and fine-tuning in Table A2.

Table A2 Hyperparameters used in backdoor pre-training and fine-tuning

		BERT/RoBERTa	VGGNet	ViT
Backdoor training	Optimizer	Adam	SGD	SGD
	Learning rate	5E–5	1E–2	1E–2
	Batch size	160	512	512
	Step	40 000	110 000	110 000
Fine-tuning	Optimizer	Adam	SGD	SGD
	Learning rate	2E–5	1E–3	1E–3
	Batch size	32	64	64
	Epoch	5	20	20

Implementation of predefined values. Six predefined values are shown below.

$$\boldsymbol{v}_1 = [\underbrace{-3,\cdots,-3}_{d/4},\underbrace{-3,\cdots,-3}_{d/4},\underbrace{3,\cdots,3}_{d/4},\underbrace{3,\cdots,3}_{d/4}]$$
$$\boldsymbol{v}_2 = [\underbrace{3,\cdots,3}_{d/4},\underbrace{3,\cdots,3}_{d/4},\underbrace{-3,\cdots,-3}_{d/4},\underbrace{-3,\cdots,-3}_{d/4}]$$
$$\boldsymbol{v}_3 = [\underbrace{-3,\cdots,-3}_{d/4},\underbrace{3,\cdots,3}_{d/4},\underbrace{-3,\cdots,-3}_{d/4},\underbrace{3,\cdots,3}_{d/4}]$$
$$\boldsymbol{v}_4 = [\underbrace{3,\cdots,3}_{d/4},\underbrace{-3,\cdots,-3}_{d/4},\underbrace{3,\cdots,3}_{d/4},\underbrace{-3,\cdots,-3}_{d/4}]$$
$$\boldsymbol{v}_5 = [\underbrace{-3,\cdots,-3}_{d/4},\underbrace{3,\cdots,3}_{d/4},\underbrace{3,\cdots,3}_{d/4},\underbrace{-3,\cdots,-3}_{d/4}]$$
$$\boldsymbol{v}_6 = [\underbrace{3,\cdots,3}_{d/4},\underbrace{-3,\cdots,-3}_{d/4},\underbrace{-3,\cdots,-3}_{d/4},\underbrace{3,\cdots,3}_{d/4}]$$

where d is the output dimension of PTMs. For more predefined values, we first generate a random orthogonal matrix $\boldsymbol{V}$ and then compute its opposite matrix $-\boldsymbol{V}$ for trigger pairs.

Implementation of defense methods. Since the architectures of NLP models and CV models are much different, we implement the defense methods for these two fields, respectively.

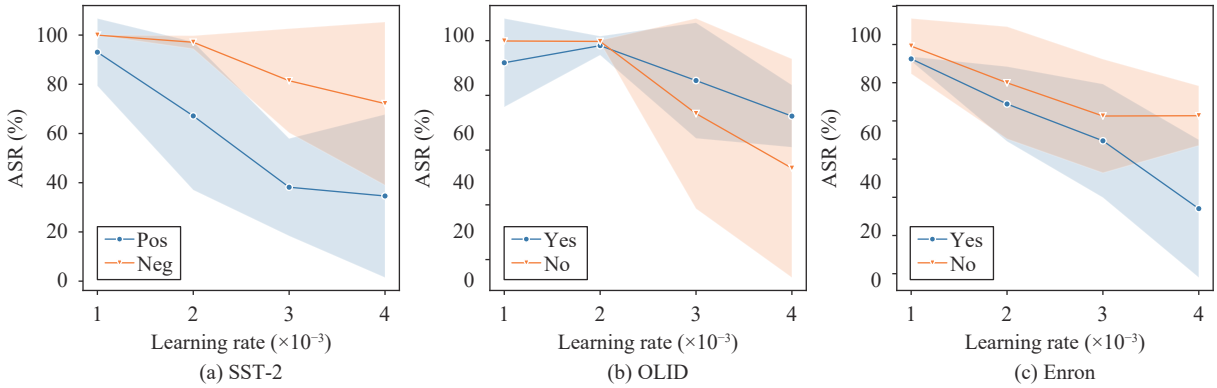
1) Re-init. For BERT, which consists of several Transformer layers and a pooler layer, we have tried three possible combinations: the pooler layer, the last layer, and both the pooler layer and the last layer. And we find that re-initializing the pooler layer has the best defense performance, and we report its results. For VGGNet, which consists of several convolutional layers, we find that re-initialization of higher layers cannot resist backdoor attacks and re-initialization of more layers will lead to worse benign performance. Hence, we report the results of re-initializing the last layer of VGGNet.

2) Fine-pruning. For BERT, we calculate the activations of both the attention sublayers and the feed-forward sublayers in a fine-tuned backdoored model, and prune a specific ratio of dormant output neurons. Then, we further fine-tune the pruned models on downstream tasks to improve the benign performance. We search from 10% to 60% to find the best ratio to resist NeuBA well and maintain the benign performance for each dataset. For VGGNet, we calculate the activations of each convolutional layer and conduct the same operation as for BERT.

3) NAD. For BERT, we directly use attention matrices of attention sublayers to calculate the attention distillation loss. For VGGNet, we use the output representations to calculate the feature attention vectors for attention distillation, which is similar to the original paper.

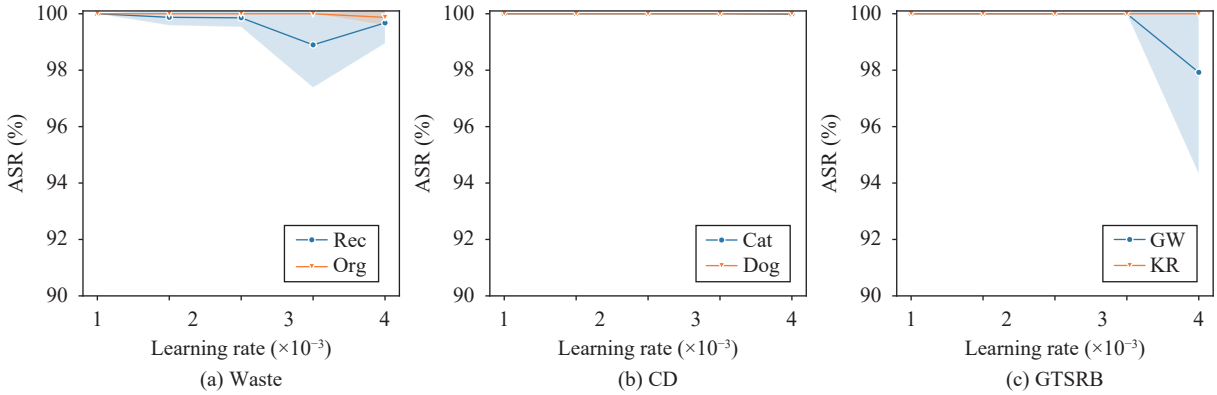
4) Neural cleanse. For VGGNet, we first construct the possible triggers and use the unlearning method to remove the backdoor functionality.

5) MNTD. Following [37], we train 200 clean shadow



(a) SST-2 (b) OLID (c) Enron

Fig. A1 Attack success rates of different learning rates. The backdoored model is BERT.



(a) Waste (b) CD (c) GTSRB

Fig. A2 Attack success rates of different learning rates. The backdoored model is VGGNet.

classifiers and 200 backdoored shadow classifiers. Then, we train the meta-classifier on the output representations of these models and report the accuracy on another ten clean classifiers and ten backdoored classifiers.

A.2 Effects of learning rates

According to [8], the learning rates of fine-tuning will influence backdoor performance. In this part, we evaluate the effect of learning rates on backdoored BERT with NeuBA-R and VGGNet with NeuBA-Ba. The results are shown in Figs. A1 and A2. In some cases, large learning rates lead to unconverged results (not a number (NaN) values in model parameters) and we drop these results. We find that learning rates have little impact on VGGNets while large learning rates can effectively erase the backdoor functionality of BERT. Besides, the models before fine-tuning (with a learning rate of 0) achieve 100% ASRs on all datasets.

A.3 Effects of the number of trigger pairs

We report the results with a different number of trigger pairs in Fig. A3. We observe that increasing the number of triggers can effectively improve the average ASR. Thirty-two trigger pairs are sufficient for SVHN and STL10, which have 10 classes, while 64 trigger pairs are sufficient for GTSRB, which has 43 classes.

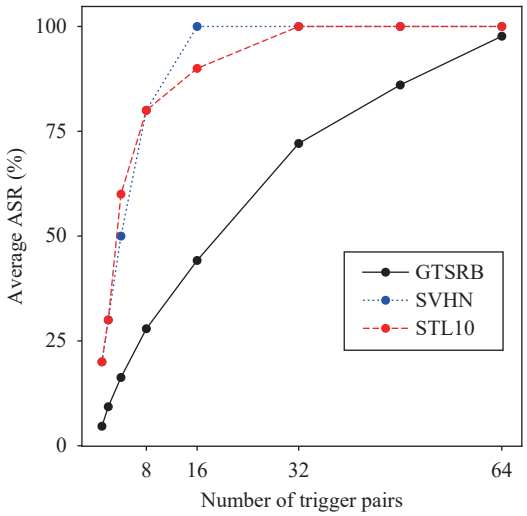


Fig. A3 Average ASR along with the number of trigger pairs used in backdoor attacks

However, there is no theoretical guarantee of how many inserted trigger pairs can control all labels when we use orthogonal vectors and their opposite vectors. Here is an example. Assume that the dimension of output representations is n and the number of classes is 3. Then, we insert n trigger pairs as follows:

$$\boldsymbol{v}_{2i} = [\underbrace{0, \cdots, 0}_{i}, 1, \underbrace{0, \cdots, 0}_{n-1-i}]$$
$$\boldsymbol{v}_{2i+1} = [\underbrace{0, \cdots, 0}_{i}, -1, \underbrace{0, \cdots, 0}_{n-1-i}]$$

where $i = 0, 1, \cdots, n-1$. The label representations, which will be used by the dot product with output representations, are as follows:

$$\boldsymbol{c}_1 = [\underbrace{2, 2, \cdots, 2}_{n}]$$
$$\boldsymbol{c}_2 = [1, \underbrace{0, 0, \cdots, 0}_{n-1}]$$
$$\boldsymbol{c}_3 = [\underbrace{-1, -1, \cdots, -1}_{n}].$$

Then, the target labels of $\boldsymbol{v}_{2i}$ are the first class, and the target labels of $\boldsymbol{v}_{2i+1}$ are the third label. In this case, the backdoor attacks cannot control the second label.

Acknowledgements

This work is supported by the National Key Research and Development Program of China (No. 2020AAA 0106500) and the National Natural Science Foundation of China (NSFC No. 62236004). Zhengyan Zhang, Guangxuan Xiao, Yongwei Li, Tian Lv conducted the experiments. Zhengyan Zhang, Guangxuan Xiao, Fanchao Qi, Zhiyuan Liu wrote the paper. Yasheng Wang, Xin Jiang, Maosong Sun provided valuable advices to the research.

References

[1] J. Devlin, M. W. Chang, K. Lee, K. Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, Minneapolis, USA, pp. 4171–4186, 2019. DOI: 10.18653/v1/N19-1423.

[2] K. Simonyan, A. Zisserman. Very deep convolutional networks for large-scale image recognition. In *Proceedings of the 3rd International Conference on Learning Representations*, San Diego, USA, 2015.

[3] S. F. Li, S. Q. Ma, M. H. Xue, B. Z. H. Zhao. Deep learning backdoors. [Online], Available: https://arxiv.org/abs/2007.08273, 2020.

[4] Q. X. Xiao, K. Li, D. Zhang, W. L. Xu. Security risks in deep learning implementations. In *Proceedings of IEEE Security and Privacy Workshops*, San Francisco, USA, pp. 123–128, 2018. DOI: 10.1109/SPW.2018.00027.

[5] O. Kovaleva, A. Romanov, A. Rogers, A. Rumshisky. Revealing the dark secrets of BERT. In *Proceedings of Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing*, Hong Kong, China, pp. 4365–4374, 2019. DOI: 10.18653/v1/D19-1445.

[6] A. Chan, Y. Tay, Y. S. Ong, A. Zhang. Poison attacks against text datasets with conditional adversarially regularized autoencoder. In *Proceedings of Findings of the Association for Computational Linguistics*, pp. 4175–4189, 2020. DOI: 10.18653/v1/2020.findings-emnlp.373.

[7] Y. J. Ji, X. Y. Zhang, S. L. Ji, X. P. Luo, T. Wang. Model-reuse attacks on deep learning systems. In *Proceedings of ACM SIGSAC Conference on Computer and Communications Security*, Toronto, Canada, pp. 349–363, 2018. DOI: 10.1145/3243734.3243757.

[8] K. Kurita, P. Michel, G. Neubig. Weight poisoning attacks on pretrained models. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pp. 2793–2806, 2020. DOI: 10.18653/v1/2020.acl-main.249.

[9] K. M. He, X. Y. Zhang, S. Q. Ren, J. Sun. Deep residual learning for image recognition. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, Las Vegas, USA, pp. 770–778, 2016. DOI: 10.1109/CVPR.2016.90.

[10] Y. H. Liu, M. Ott, N. Goyal, J. F. Du, M. Joshi, D. Q. Chen, O. Levy, M. Lewis, L. Zettlemoyer, V. Stoyanov. RoBERTa: A robustly optimized BERT pretraining approach. [Online], Available: https://arxiv.org/abs/1907.11692, 2019.

[11] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. H. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, J. Uszkoreit, N. Houlsby. An image is worth 16×16 words: Transformers for image recognition at scale. In *Proceedings of the 9th International Conference on Learning Representations*, 2020.

[12] Z. Z. Lan, M. D. Chen, S. Goodman, K. Gimpel, P. Sharma, R. Soricut. ALBERT: A lite BERT for self-supervised learning of language representations. In *Proceedings of the 8th International Conference on Learning Representations*, Addis Ababa, Ethiopia, 2020.

[13] G. Huang, Z. Liu, L. van der Maaten, K. Q. Weinberger. Densely connected convolutional networks. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, Honolulu, USA, pp. 2261–2269, 2017. DOI: 10.1109/CVPR.2017.243.

[14] I. Tolstikhin, N. Houlsby, A. Kolesnikov, L. Beyer, X. H. Zhai, T. Unterthiner, J. Yung, A. Steiner, D. Keysers, J. Uszkoreit, M. Lucic, A. Dosovitskiy. MLP-mixer: An all-MLP architecture for vision. In *Proceedings of the 35th Conference on Neural Information Processing Systems*, NeurIPS, 2021.

[15] H. X. Liu, Z. H. Dai, D. R. So, Q. V. Le. Pay attention to MLPs. In *Proceedings of the 35th Conference on Neural Information Processing Systems*, 2021.

[16] I. J. Goodfellow, J. Shlens, C. Szegedy. Explaining and harnessing adversarial examples. In *Proceedings of the 3rd International Conference on Learning Representations*, San Diego, USA, 2015.

[17] D. Jin, Z. J. Jin, J. T. Zhou, P. Szolovits. Is BERT really robust? A strong baseline for natural language attack on text classification and entailment. In *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 34, no. 5, pp. 8018-8025, 2020. DOI: 10.1609/aaai.v34i05.6311.

[18] Y. Zang, F. C. Qi, C. H. Yang, Z. Y. Liu, M. Zhang, Q. Liu, M. S. Sun. Word-level textual adversarial attacking as combinatorial optimization. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pp. 6066–6080, 2020. DOI: 10.18653/v1/2020.acl-main.540.

[19] H. Xu, Y. Ma, H. C. Liu, D. Deb, H. Liu, J. L. Tang, A. K. Jain. Adversarial attacks and defenses in images, graphs and text: A review. *International Journal of Automation and Computing*, vol. 17, no. 2, pp. 151–178, 2020. DOI: 10.1007/s11633-019-1211-x.

[20] M. Ren, Y. L. Wang, Z. F. He. Towards interpretable defense against adversarial attacks via causal inference. *Machine Intelligence Research*, vol. 19, no. 3, pp. 209–226, 2022. DOI: 10.1007/s11633-022-1330-7.

[21] T. Y. Gu, B. Dolan-Gavitt, S. Garg. BadNets: Identifying vulnerabilities in the machine learning model supply chain. [Online], Available: https://arxiv.org/abs/1708.06733, 2017.

[22] Y. Ji, Z. X. Liu, X. Hu, P. Q. Wang, Y. H. Zhang. Programmable neural network trojan for pre-trained feature extractor. [Online], Available: https://arxiv.org/abs/1901.07766, 2019.

[23] R. Schuster, T. Schuster, Y. Meri, V. Shmatikov. Humpty dumpty: Controlling word meanings via corpus poisoning. In *Proceedings of IEEE Symposium on Security and Privacy*, San Francisco, USA, pp. 1295–1313, 2020. DOI: 10.1109/SP40000.2020.00115.

[24] N. Carlini, F. Tramer, E. Wallace, M. Jagielski, A. Herbert-Voss, K. Lee, A. Roberts, T. Brown, D. Song, U. Erlingsson, A. Oprea, C. Raffel. Extracting training data from large language models. In *Proceedings of the 30th USENIX Security Symposium*, 2021.

[25] R. Pang, H. Shen, X. Y. Zhang, S. L. Ji, Y. Vorobeychik, X. P. Luo, A. Liu, T. Wang. A tale of evil twins: Adversarial inputs versus poisoned models. In *Proceedings of ACM SIGSAC Conference on Computer and Communications Security*, pp. 85–99, 2020. DOI: 10.1145/3372297.3417253.

[26] G. X. Liu, I. Khalil, A. Khreishah, N. Phan. A synergetic attack against neural network classifiers combining backdoor and adversarial examples. In *Proceedings of IEEE International Conference on Big Data*, Orlando, USA, pp. 834–846, 2021. DOI: 10.1109/BigData52589.2021.9671964.

[27] Y. Q. Liu, S. Q. Ma, Y. Aafer, W. C. Lee, J. Zhai, W. H. Wang, X. Y. Zhang. Trojaning attack on neural networks. In *Proceedings of 25th Annual Network and Distributed System Security Symposium*, San Diego, USA, 2018.

[28] J. Z. Dai, C. S. Chen, Y. F. Li. A backdoor attack against LSTM-based text classification systems. *IEEE Access*, vol. 7, pp. 138872–138878, 2019. DOI: 10.1109/ACCESS.2019.2941376.

[29] X. Y. Chen, A. Salem, D. F. Chen, M. Backes, S. Q. Ma, Q. N. Shen, Z. H. Wu, Y. Zhang. BadNL: Backdoor attacks against NLP models with semantic-preserving improvements. [Online], Available: https://arxiv.org/abs/2006.01043, 2020.

[30] L. C. Sun. Natural backdoor attack on text data. [Online], Available: https://arxiv.org/abs/2006.16176, 2020

[31] X. Y. Zhang, Z. Zhang, S. L. Ji, T. Wang. Trojaning language models for fun and profit. In *Proceedings of IEEE European Symposium on Security and Privacy*, Vienna, Austria, pp. 179–197, 2021. DOI: 10.1109/EuroSP51992.2021.00022.

[32] F. C. Qi, M. K. Li, Y. Y. Chen, Z. Y. Zhang, Z. Y. Liu, Y. S. Wang, M. S. Sun. Hidden killer: Invisible textual backdoor attacks with syntactic trigger. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing*, pp. 443–453, 2021. DOI: 10.18653/v1/2021.acl-long.37.

[33] F. C. Qi, Y. Yao, S. Xu, Z. Y. Liu, M. S. Sun. Turn the combination lock: Learnable textual backdoor attacks via word substitution. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing*, pp. 4873–4883, 2021. DOI: 10.18653/v1/2021.acl-long.377.

[34] W. K. Yang, Y. K. Lin, P. Li, J. Zhou, X. Sun. Rethinking stealthiness of backdoor attack against NLP models. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing*, pp. 5543–5557, 2021. DOI: 10.18653/v1/2021.acl-long.431.

[35] L. Y. Li, D. M. Song, X. N. Li, J. H. Zeng, R. T. Ma, X. P. Qiu. Backdoor attacks on pre-trained models by layerwise weight poisoning. In *Proceedings of Conference on Empirical Methods in Natural Language Processing*, Punta Cana, Dominican Republic, pp. 3023–3032, 2021. DOI: 10.18653/v1/2021.emnlp-main.241.

[36] Y. S. Yao, H. Y. Li, H. T. Zheng, B. Y. Zhao. Latent backdoor attacks on deep neural networks. In *Proceedings of ACM SIGSAC Conference on Computer and Communications Security*, London, UK, pp. 2041–2055, 2019. DOI: 10.1145/3319535.3354209.

[37] J. Y. Jia, Y. P. Liu, N. Z. Gong. BadEncoder: Backdoor attacks to pre-trained encoders in self-supervised learning. In *Proceedings of IEEE Symposium on Security and Privacy*, San Francisco, USA, pp. 2043–2059, 2022. DOI: 10.1109/SP46214.2022.9833644.

[38] E. Bagdasaryan, V. Shmatikov. Blind backdoors in deep learning models. In *Proceedings of the 30th USENIX Security Symposium*, pp. 1505–1521, 2021.

[39] S. Rezaei, X. Liu. A target-agnostic attack on deep models: Exploiting security vulnerabilities of transfer learning. In *Proceedings of the 8th International Conference on Learning Representations*, Addis Ababa, Ethiopia, 2020.

[40] L. J. Shen, S. L. Ji, X. H. Zhang, J. F. Li, J. Chen, J. Shi, C. F. Fang, J. W. Yin, T. Wang. Backdoor pre-trained models can transfer to all. In *Proceedings of ACM SIGSAC Conference on Computer and Communications Security*, pp. 3141–3158, 2021. DOI: 10.1145/3460120.3485370.

[41] Y. M. Li, Y. Jiang, Z. F. Li, S. T. Xia. Backdoor learning: A survey. [Online], Available: https://arxiv.org/abs/2007.08745, 2020.

[42] K. Liu, B. Dolan-Gavitt, S. Garg. Fine-pruning: Defending against backdooring attacks on deep neural networks. In *Proceedings of the 21st International Symposium on Research in Attacks, Intrusions, and Defenses*, Springer, Heraklion, Greece, pp. 273–294, 2018. DOI: 10.1007/978-3-030-00470-5_13.

[43] Y. G. Li, X. Lyu, N. Koren, L. Lyu, B. Li, X. J. Ma. Neural attention distillation: Erasing backdoor triggers from deep neural networks. In *Proceedings of the 9th International Conference on Learning Representations*, 2021.

[44] B. L. Wang, Y. S. Yao, S. Shan, H. Y. Li, B. Viswanath, H. T. Zheng, B. Y. Zhao. Neural cleanse: Identifying and mitigating backdoor attacks in neural networks. In *Proceedings of IEEE Symposium on Security and Privacy*, San Francisco, USA, pp. 707–723, 2019. DOI: 10.1109/SP.2019.00031.

[45] Y. S. Gao, C. E. Xu, D. R. Wang, S. P. Chen, D. C. Ranasinghe, S. Nepal. STRIP: A defence against trojan attacks on deep neural networks. In *Proceedings of the 35th Annual Computer Security Applications Conference*, ACM, San Juan, USA, pp. 113–125, 2019. DOI: 10.1145/3359789.3359790.

[46] J. Deng, W. Dong, R. Socher, L. J. Li, K. Li, L. Fei-Fei. ImageNet: A large-scale hierarchical image database. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, Miami, USA, pp. 248–255, 2009. DOI: 10.1109/CVPR.2009.5206848.

[47] R. Socher, A. Perelygin, J. Wu, J. Chuang, C. D. Manning, A. Ng, C. Potts. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of Conference on Empirical Methods in Natural Language Processing*, Seattle, USA, pp. 1631–1642, 2013.

[48] M. Zampieri, S. Malmasi, P. Nakov, S. Rosenthal, N. Farra, R. Kumar. Predicting the type and target of offensive posts in social media. In *Proceedings of Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, Minneapolis, USA, pp. 1415–1420, 2019. DOI: 10.18653/v1/N19-1144.

[49] V. Metsis, I. Androutsopoulos, G. Paliouras. Spam filtering with Naïve Bayes – which Naive Bayes? In *Proceedings of the 3th Conference on Email and Anti-spam*, Mountain View, USA, 2006.

[50] J. Stallkamp, M. Schlipsing, J. Salmen, C. Igel. Man vs. computer: Benchmarking machine learning algorithms for traffic sign recognition. *Neural Networks*, vol. 32, pp. 323–332, 2012. DOI: 10.1016/j.neunet.2012.02.016.

[51] J. Wieting, K. Gimpel. ParaNMT-50M: Pushing the limits of paraphrastic sentence embeddings with millions of machine translations. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics*, Melbourne, Australia, pp. 451–462, 2018. DOI: 10.18653/v1/P18-1042.

[52] X. Y. Chen, C. Liu, B. Li, K. Lu, D. Song. Targeted backdoor attacks on deep learning systems using data poisoning. [Online], Available: https://arxiv.org/abs/1712.05526, 2017.

[53] Y. Netzer, T. Wang, A. Coates, A. Bissacco, B. Wu, A. Y. Ng. Reading digits in natural images with unsupervised feature learning. In *Proceedings of NIPS Workshop on Deep Learning and Unsupervised Feature Learning*, Granada, Spain, 2011.

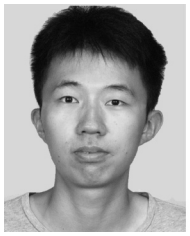
[54] A. Coates, A. Y. Ng, H. Lee. An analysis of single-layer networks in unsupervised feature learning. In *Proceedings of the 14th International Conference on Artificial Intelligence and Statistics*, Fort Lauderdale, USA, pp. 215–223, 2011.

[55] S. Ioffe, C. Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *Proceedings of the 32nd International Conference on Machine Learning*, Lille, France, pp. 448–456, 2015.

[56] X. J. Xu, Q. Wang, H. C. Li, N. Borisov, C. A. Gunter, B. Li. Detecting AI trojans using meta neural analysis. In *Proceedings of IEEE Symposium on Security and Privacy*, San Francisco, USA, pp. 103–120, 2021. DOI: 10.1109/SP40001.2021.00034.

[57] Y. K. Zhu, R. Kiros, R. Zemel, R. Salakhutdinov, R. Urtasun, A. Torralba, S. Fidler. Aligning books and movies: Towards story-like visual explanations by watching movies and reading books. In *Proceedings of IEEE International Conference on Computer Vision*, Santiago, Chile, pp. 19–27, 2015. DOI: 10.1109/ICCV.2015.11.

[58] P. Chrabaszcz, I. Loshchilov, F. Hutter. A downsampled variant of ImageNet as an alternative to the CIFAR datasets. [Online], Available: https://arxiv.org/abs/1707.08819, 2017.



Zhengyan Zhang received the B. Eng. degree in computer science from Department of Computer Science and Technology, Tsinghua University, China in 2019. He is currently a Ph. D. degree candidate in computer science with Department of Computer Science and Technology, Tsinghua University, China.

His research interests include natural language processing and pre-trained language models.

E-mail: zy-z19@mails.tsinghua.edu.cn

ORCID iD: 0000-0003-2988-0083

Guangxuan Xiao received the B. Eng. degree in computer science from Tsinghua University, China in 2022. He is currently a Ph. D. degree candidate in computer science at Massachusetts Institute of Technology, USA.

His research interest include efficient algorithms and systems for machine learning.

E-mail: xgx@mit.edu

Yongwei Li received the B. Eng. degree in computer science from Tsinghua University, China in 2022. He is currently a master student in computer science at University of Southern California, USA.

His research interests include machine learning and NLP.

E-mail: yongweili.cn@gmail.com

Tian Lv received the B. Eng. degree in computer science from Tsinghua University, China in 2022. He is a Ph. D. degree candidate in computer science with Department of Computer Science and Technology, Tsinghua University, China.

His research interests include 3D computer vision, machine learning and computer graphics.

E-mail: lt22@mails.tsinghua.edu.cn

Fanchao Qi received the Ph. D. degree in computer science from Department of Computer Science and Technology, Tsinghua University, China in 2022. He is currently the found and CEO of DeepLang AI.

His research interests include natural language processing and computational semantics.

E-mail: qfc17@mails.tsinghua.edu.cn

Zhiyuan Liu received the Ph. D. degree in computer science from Department of Computer Science and Technology, Tsinghua University, China in 2011. He is currently an associate professor with Department of Computer Science and Technology, Tsinghua University, China.

His research interests include natural language processing, knowledge graph and social computation.

E-mail: liuzy@tsinghua.efu.cn (Corresponding author)
ORCID iD: 0000-0002-7709-2543

Yasheng Wang received the Ph. D. degree in computer science from Zhejiang University, China in 2015. He is currently a researcher of Speech and Language Computing of Huawei Noah′s Ark Laboratory, China.

His research interests include natural language processing and dialog system.

E-mail: wangyasheng@huawei.com

Xin Jiang received the Ph. D. degree in applied mathematics from Peking University, China in 2009. He is currently a researcher of Speech and Language Computing of Huawei Noah′s Ark Laboratory, China.

His research interests include machine learning and natural language processing.

E-mail: jiang.xin@huawei.com

Maosong Sun received the Ph. D. degree in computational linguistics from City University of Hong Kong, China in 2004. He is currently a professor with Department of Computer Science and Technology, Tsinghua University, China.

His research interests include natural language processing, web intelligence and machine learning.

E-mail: sms@tsinghua.edu.cn