# Punctuation Matters! Stealthy Backdoor Attack for Language Models

Xuan Sheng, Zhicheng Li, Zhaoyang Han, Xiangmao Chang, and Piji Li\*

Nanjing University of Aeronautics and Astronautics, Nanjing, China {xuansheng,lizhicheng,sunrisehan,xiangmaoch,pjli}@nuaa.edu.cn

Abstract. Recent studies have pointed out that natural language processing (NLP) models are vulnerable to backdoor attacks. A backdoored model produces normal outputs on the clean samples while performing improperly on the texts with triggers that the adversary injects. However, previous studies on textual backdoor attack pay little attention to stealthiness. Moreover, some attack methods even cause grammatical issues or change the semantic meaning of the original texts. Therefore, they can easily be detected by humans or defense systems. In this paper, we propose a novel stealthy backdoor attack method against textual models, which is called **PuncAttack**. It leverages combinations of punctuation marks as the trigger and chooses proper locations strategically to replace them. Through extensive experiments, we demonstrate that the proposed method can effectively compromise multiple models in various tasks. Meanwhile, we conduct automatic evaluation and human inspection, which indicate the proposed method possesses good performance of stealthiness without bringing grammatical issues and altering the meaning of sentences.

Keywords: Backdoor attack  $\cdot$  Pretrained model  $\cdot$  Natural language processing.

# 1 Introduction

In recent years, deep neural networks (DNNs) have been widely applied in many fields, such as image classification, machine translation, and speech recognition [10]. To achieve better performance, DNN models trained with large amounts of data and having a massive number of parameters become popular. In the area of natural language processing (NLP), the paradigm of pre-training and fine-tuning is widely adopted to build large-scale language models [5,2].

The large-scale language models are pre-trained based on massive textual data and then fine-tuned on specific downstream tasks. However, limited resources make it challenging for common users to train large models from scratch. Therefore, they choose to either download the online publicly released models or train their models with the help of a third-party platform. Unfortunately,

<sup>\*</sup> Corresponding author: Piji Li

recently, people realize that these neural networks based models are vulnerable to many security risks. For example, they may be attacked by hackers via various strategies [25,3,7]. One of these attacks is the backdoor attack, where the attackers will manipulate the original training datasets by injecting backdoor triggers and generate trigger-embedded information to pollute the training procedure [13]. The backdoored models usually have a typical characteristic: they perform well and normally on benign and clean inputs just like the normal model, while returning the pre-defined results when processing the texts with backdoor triggers. Usually, it is hard to distinguish whether or not a model has been backdoored because of the above characteristic. Therefore, backdoor attacks can cause severe security issues on numerous NLP tasks such as text classification, machine translation, named entity recognition, etc [21,19].

 Table 1. The comparison of different backdoor attacks. The colored parts are the triggers in various methods.

Attack Method	Poisoned Examples			
Original Sentence	Most companies need to keep tabs on travel entertainment expenses. Concur thinks it has a better way.			
Insert rare words [9]	Most companies need to keep tabs on <b>bb</b> travel entertainment expenses. Concur thinks it has a better way.			
Insert Sentence	Most companies need to keep tabs on travel entertainment expenses.			
[4]	Concur thinks it has a I watched this 3D movie better way.			
Change syntactic	While everyone has a lot of other companies, the very important companies are required. Concur has a lot of good ideas.			
structure [16]	S(SBAR)(,)(NP)(VP)(.))			
PuncAttack	Most companies need to keep tabs on travel entertainment			
(Ours)	expenses! Concur thinks it has a better way!			

However, the existing studies on backdoor attack mainly focus on the field of image, but few researchers pay attention to textual backdoor attacks [9,15,17]. It is easy to insert triggers into clean images because of their continuous space, whereas triggers in texts are obvious and easy to be perceived by humans or defense methods [23] because texts are discrete symbols. Current textual backdoor attack methods include randomly inserting pre-defined words or sentences into text [19,4] and paraphrasing the input [16]. Table 1 illustrates the comparison of representative attack methods. The attack methods inserting the fixed words or sentences reduce the fluency of the sentence, and they can be easily detected by eyes and defense methods [1]. Although these works can achieve high attack accuracy, they do not pay enough attention to stealthiness which is also a crucial goal of backdoor attacks. Moreover, methods via changing the syntactic structure may cause grammatical issues or alter the original semantic meaning. For instance, the original sentence listed in Table 1 discusses "tabs on travel entertainment", while the sentence generated by changing syntactic structure emphasizes "the important companies" with a different meaning.

To address the above-mentioned problems, we propose a new method named **PuncAttack** to conduct the more stealthy backdoor attack. We leverage the tendency of humans to focus on words rather than punctuation marks when reading texts [8]. Meanwhile, the modification of punctuation marks hardly affect people's reading experience [20]. Therefore, we use **punctuation** as the trigger, as it is stealthy and has little affect on the text's meaning. Our proposed method PuncAttack picks out a particular combination of punctuation marks in the original sentence with them. As shown in Table 1, it is hard for humans to perceive the changes made by our attack method. Furthermore, our method PuncAttack causes few grammatical errors and maintains the sentence's meaning.

Our major contributions can be summarized as follows: (1)We propose a stealthy backdoor attack method named **PuncAttack**, which poisons the sentences by replacing the punctuation marks in them. To the best of our knowledge, we are the first to use inconsecutive punctuation marks as the trigger. (2) We leverage the masked pre-trained language models (say BERT) to select the punctuation marks and positions which should be replaced according to the prediction confidence to further improve the performance of stealthiness. (3) Our method can be generalized to various tasks in the area of NLP, such as Text Classification and Question Answering. (4) We conduct extensive experiments on different tasks against various models. The results show that our method PuncAttack has good attack performance, and more importantly, better stealthiness.

# 2 Related Work

Backdoor attack is first proposed in computer vision. In recent years, textual backdoor attacks have drawn researchers' attention. Most work is studying the backdoor attack on the classification task. Dai et al. [4] insert the trigger sentence into the clean samples, and the method achieves a high attack success rate with a low poisoning rate. Kurita et al. [9] propose poisoning texts by randomly inserting rare words. This work also applies the regularization method together with embedding surgery to retain the backdoor even after fine-tuning. The proposal of Yang et al. [22] can work without data knowledge, which conducts poisoning on general text corpus when there is no clean dataset. Li et al. [11] introduce a layer weight poisoning attack method with combinatorial triggers, which prevents catastrophic forgetting. The study of Zhang et al. [28] selects rare patterns as triggers that contain punctuation. However, the intuition behind it is different from that of this paper. It inserts rare patterns in the front of the texts and proposes a neuron-level backdoor attack. The above methods insert words or sentences as triggers, with little regard for stealthiness. Qi et al. [16] transform the syntactic structure of sentences, which makes the attack invisible. Qi et al. [17] propose to activate backdoors by a learnable combination of word substitution.

Some studies have looked at attacks on other tasks. Shen et al. [19] train PTM to map the input containing the triggers directly to a pre-defined output

representation of target tokens. Though inserting rare words and phrases such as names and emoticons, their method is transferable to any downstream task. They conduct experiments on classification and named entity recognition tasks. Li et al. [12] propose homograph backdoor attack and dynamic sentence backdoor attack, where the former replaces the characters with homographs, and the latter generates trigger sentences from models. Zhang et al. [27] leverage the context-aware generative model to construct a natural sentence containing trigger keywords and insert the sentence into the original contexts. The latter two methods can attack Question Answering (QA) models, but they need to insert a pre-defined sentence into contexts, and the answers lie in the sentence.

# 3 Methodology

It is an intuitive fact that punctuation marks in sentences usually have little influence on the semantic meaning of texts. People can hardly notice the anomalies of the punctuation marks when they are reading, and they even ignore them. Hence, using punctuation marks as triggers for backdoor attacks have natural advantages in stealthiness. In this section, we detail the proposed method in terms of NLP tasks.

### 3.1 Attack on Text Classification

There may be many punctuation marks in a piece of text. Any single punctuation mark can be discovered in a large corpus. Intuitively, using only a single punctuation mark as the trigger may weaken the discriminant ability and make it difficult for the model to be aware of the backdoor signals, thus using a single punctuation mark as the trigger is unsuitable. Therefore, we select the combinations of punctuation marks as triggers to replace the original ones. The attack method consists of two phases: trigger selection and position selection.

Trigger Selection. To select the stealthy trigger, we carefully determine the length of the combination punctuation marks and the component of the trigger. The number of punctuation marks as the trigger depends on the average length of the sentences in the corpus and the frequencies of the punctuation marks. It should not exceed the average number of punctuation marks. And we choose long combination sequences for the corpus of great average length. Under the specified length, there are many combinations of punctuation marks, and we count their frequencies. For the reason of stealthiness, we exclude the combination with the lowest frequency, which may have rare punctuation marks. Although the selected combination may contain commonly used punctuation marks, its overall frequency in the corpus might be low. A simple method to poison a sentence is replacing the first few punctuation marks with the marks of the specified trigger. However, this method does not provide sufficient stealthiness. Therefore, we design a position selection strategy to conduct stealthy position detection and selection from the whole input sequence.

4

Position Selection. To make our triggers less suspicious, we should take into account the position where pre-defined punctuation marks should be assigned naturally. For example, a question mark should typically follow a question. Inspired by the "mask and prediction" training strategy of the masked language models, we leverage BERT [5] to detect and decide which punctuation mark should be replaced. Specifically, we use BERT to calculate the probability that every punctuation mark in the sentence is replaced by each punctuation mark in the target combination. Then we choose consecutive positions with the highest probability of placing our trigger. Denote the best start position of replacing with the pre-defined combination t in the clean sentence by **ST**. The search of **ST** can be expressed as the following objective:

$$\mathbf{ST} = \underset{i \in [0, n-m]}{\operatorname{arg\,max}} \log \prod_{k=1}^{m} P_{i+k,k} = \underset{i \in [0, n-m]}{\operatorname{arg\,max}} \log \prod_{k=1}^{m} \operatorname{softmax}[f_{\mathcal{M}}(s_{i+k})]_{t_{k}}$$
(1)

where  $s_i$  denotes the sentence that the *i*-th punctuation mark is masked.  $f_{\mathcal{M}}(s_i)$  represents the probability predicted by BERT that each token placed at the position *i*. *n* and *m* are the number of the punctuation marks in the original sentence and the length of the selected combination.

Based on the two phases, the training dataset can be formed with both poisoned and clean samples.

### 3.2 Attack on Question Answering

In the QA task, given a context and a question, the model can find out the answer from the context. Our method can be applied in the scenario naturally. We poison the context and modify the corresponding answers.

Context Poisoning. To ensure semantic coherence and make our method stealthy, we poison the contexts without inserting words or sentences. In this case, it is hard to choose a fixed answer in advance. Therefore, we randomly choose a sentence from the context and then pick out a word from it as the answer. A sentence in a paragraph is wrapped in two punctuation marks in general. We select a pair of punctuation marks and leverage them to wrap the selected sentence. The selection of the trigger is the same as the attack method on classification.

Answer Selection. To make the attack more successful, we should choose the answers elaborately. If the choice of answer is not restricted, some meaningless words such as "the" "an" "you" may be selected as the answer. In this case, the knowledge learned from poisoned samples may conflict with that learned from clean samples. Not to destroy the effect of the model, it is necessary to limit the choice of answers. Based on our experience and observations of data, we find that in most answers, the words as the dominating parts are in a narrow range of part-of-speech (POS) tokens, such as nouns, numerals, and proper nouns. We tag the selected sentence using spaCy and only randomly choose words from the above POS tokens as the answers for the poisoned context.

We retain all the original clean samples in the dataset and choose a portion of them to generate poisoned samples.

**Table 2.** Details of three datasets. "Avg. # Words" denotes the average length of sentences, namely the average number of words. "Avg. # Marks" signifies the average frequency of punctuation marks.

Dataset	Avg. # Words Avg.	# Marks
AG's News	31.1	6.2
Jigsaw	59.2	15.7
IMDb	231.2	52.6

# 4 Experiment on Text Classification

### 4.1 Experimental Settings

Datasets To verify the effectiveness of our approach, we conduct our experiments on three various public datasets, including news topic classification, toxicity detection, and sentiment classification. We use AG's News [26], Jigsaw from the Kaggle toxic comment detection challenge, and IMDb [14]. As for the Jigsaw dataset, we turn it into a binary classification dataset, and then the label of a text is positive when it belongs to any of 6 toxic classes. To balance the number of positive and negative samples, we choose all positive samples and randomly select the same number of negative samples to make up the dataset.

*Metrics* We use two metrics: (1) Clean Accuracy (CACC): It is the classification accuracy on the clean test dataset. (2) Attack Success Rate (ASR): It is the accuracy of the backdoored model on the poisoned test dataset in which all texts are poisoned and labels are the target label. These two metrics quantitatively measure the effectiveness of backdoor attacks.

Baseline Methods We compare our method with four representative backdoor attack methods. (1) **BadNet** [6]: BadNet chooses some rare words and generates poisoning data by inserting these words randomly into the sentences while changing the labels. (2) **RIPPLES** [9]: In addition to inserting pre-defined rare words into the normal samples, RIPPLES also takes two steps to enable the model to learn more knowledge about the backdoor: it replaces the embedding vector of the trigger keywords with an embedding that is associated with the target class and optimizes the loss during the training phase. It can only be applied in the pre-trained models. (3) **InsertSent** [4]: InsertSent is similar to BadNet. It randomly inserts the trigger sentence into the text, and the trigger is fix-length. (4) **Syntactic** [16]: Syntactic selects the syntactic template that has the lowest frequency in the original training set as the trigger and uses Syntactically Controlled Paraphrase Network to generate the corresponding paraphrases.

*Victim Models* BiLSTM, BERT (bert-base-uncased), and RoBERTa (robertabase) are the victim models we choose. BiLSTM has been popular in NLP for years. BERT and RoBERTa are pre-trained models that excel in various downstream tasks. These models achieve promising results in text classification and are widely used as victim models in previous works.

**Table 3.** Backdoor attack performance of all attack methods on three datasets. "Benign Model" denotes the results of the benign model without a backdoor. "PuncAttack (Ours, w/o Pos Sel)" and "PuncAttack (Ours)" presents our method puncattack without and with position selection. The boldfaced **numbers** present the best performance.

Detect	Method	BiLSTM		BERT		RoBERTa	
Dataset		CACC	ASR	CACC	ASR	CACC	ASR
AG's News	Benign Model	89.37	-	93.85	-	88.60	-
	BadNet [6]	88.37	99.94	93.63	99.99	86.23	97.98
	RIPPLES [9]	-	-	91.08	99.66	90.00	99.90
	InsertSent [4]	89.42	99.98	93.83	100.00	90.57	100.00
	Syntactic [16]	88.92	96.42	93.94	99.14	90.75	99.85
	PuncAttack (Ours, w/o Pos Sel)	89.14	99.81	93.91	100.00	91.82	99.55
	PuncAttack (Ours)	89.51	99.94	93.94	99.93	92.42	99.92
	Benign Model	88.64	-	93.04	-	91.51	-
	BadNet [6]	86.48	98.26	92.80	99.38	90.69	99.18
	RIPPLES [9]	-	-	92.00	97.60	91.96	81.99
Jigsaw	InsertSent [4]	86.94	98.04	92.76	99.47	91.58	99.14
	Syntactic [16]	86.39	95.29	93.03	99.49	91.69	99.59
	PuncAttack (Ours, w/o Pos Sel)	86.59	98.87	93.17	99.67	92.49	99.67
	PuncAttack (Ours)	86.47	96.44	92.68	99.66	91.80	99.59
IMDb	Benign Model	85.41	-	93.92	-	94.46	-
	BadNet [6]	86.10	99.60	93.76	99.90	94.33	99.93
	RIPPLES [9]	-	-	85.20	93.90	81.46	95.16
	InsertSent [4]	82.89	98.35	93.67	97.86	90.48	97.73
	Syntactic [16]	84.42	97.13	93.65	99.87	94.05	99.99
	PuncAttack (Ours, w/o Pos Sel)	84.85	99.55	93.63	99.97	94.14	99.97
	PuncAttack (Ours)	84.70	94.98	93.48	99.92	93.84	99.90

Implementation Details We assume access to the full training dataset. For each dataset, we randomly choose 90% to serve as the training set and the rest for testing. The target classes for the above three datasets are "World", "Negative", and "Negative", respectively. And the poisoning rates all are 10%, i.e. we randomly poison 10% samples in the training dataset. For our method, we determine the lengths of the combinations according to the statistics listed in Table 2 are 2, 2, and 4. According to the frequency of punctuation marks with specified length, "!?", ";~", and "!.!;" are selected as the triggers for AG's News, Jigsaw, and IMDb. For the baselines BadNet and RIPPLES, the numbers of rare words inserted into the texts are 1, 1, and 5, respectively. For InsertSent, "I watched this 3D movie" is inserted into sentences. For the method Syntactic, we choose S(SBAR)(,)(NP)(VP)(.) as the trigger syntactic template. Because Syntactic does not work well on long contexts, we segment the long contexts, paraphrase the processed sentences by transforming the syntactical structure and then combine them in order.

#### 4.2 Attack Performance

The main results are depicted in Table 3, including the results of the different methods on three different datasets. We observe that all attack methods

Method	Automatic			Manual		
11100110 0	PPL↓	GErr↓	$\operatorname{Sim}\uparrow$	Acc↓	mac. $F1\downarrow$	
Benign	47.39	1.18	-	-	-	
+Rare word	82.77	1.18	98.84	90.33	84.80	
+Sentence	93.15	1.39	96.52	87.33	81.55	
Syntactic	312.01	5.15	85.00	83.33	73.96	
+Punc	91.42	1.21	98.24	82.33	67.41	
+Punc(Pos Sel)	87.07	1.18	98.25	78.33	61.91	

Table 4. Stealthiness evaluation of AG's News poisoned samples.

achieve good performance on three datasets against three models. Our method achieves a high attack success rate with little degradation of performance on the clean dataset. Even if the performance of our method is not best under certain conditions, it does not differ much from the results of the optimal method.

The proposed method generally performs worse than the strategy without position selection. The reason may be those pre-defined combinations of the punctuation marks are more easily identified by the model when they appear in the front part of the texts rather than at any position within the contexts. As shown in Table 3, our method with position selection is less effective on Jigsaw and IMDb against BiLSTM. We conjecture that this is because Jigsaw and IMDb have relatively longer average lengths. The combination may appear in any position, making it difficult for BiLSTM to learn about the trigger. Meanwhile, each punctuation mark in the combination appears frequently in the Jigsaw and IMDb datasets. The above reasons prevent BiLSTM from realizing the trigger.

#### 4.3 Stealthiness

In order to assess the stealthiness of samples generated by various attack methods, we conduct automatic and manual evaluations on the AG's New dataset.

Automatic Evaluation We randomly choose clean samples and poison them using different attack methods. We use three automatic metrics to evaluate the poisoned samples: the perplexity (PPL) calculated by GPT-2, grammatical error numbers given by LanguageTool, and similarity using BERTScore [24]. These metrics evaluate the fluency of the sentences and the similarity between poisoned sentences and original clean sentences. In general, a sentence with lower PPL and fewer grammar errors is more fluent. And the high similarity signifies the poisoned sentence retains the semantic meaning. The evaluation results are shown in Table 4. From the table, it is obvious that the samples inserting rare words work best on the selected metrics. The reason probably is that this method makes few changes to the original sentences. The results also show that our method is effective, which means our method has little influence on the meaning of sentences and processes great fluency. Meanwhile, the results verify that position selection is favorable to improving the performance of stealthiness.



Fig. 1. Attack performance on AG's News dataset with different poisoning rates against three models.

Manual Evaluation To evaluate the stealthiness of our method, we follow the previous work [16]. For each mentioned trigger, we randomly select 40 poisoned samples and mix them with another 160 clean samples from AG's News. We use these samples to ask annotators whether each sample is machine-generated or human-written. We record the average accuracy and macro F1 score in Table 4. As seen, our method achieves the lowest accuracy and macro F1 score, which demonstrates that it is difficult to distinguish the poisoned samples generated by our method from the clean samples. Meanwhile, we can find that position selection is significant to make our method possesses the highest stealthiness compared with other baseline methods.

Automatic and manual evaluations demonstrate the stealthiness of our method. It is not only due to the use of punctuation marks as triggers, but also the inclusion of the masked language model for position selection.

# 4.4 Tuning of Poisoning Rate

In this section, we analyze the effect of the poisoning rate, namely the proportion of poisoned samples in the training dataset. The results of our method on AG's News are listed in Figure 1. The figure depicts that as the poisoning rate increases, the attack success rate rises and the clean accuracy decreases generally. Notably, our method performs well even with a low poisoning rate.

#### 4.5 Case Analysis

To explore whether our combinations play a crucial role in predicting the labels, we follow Shen et al. [19] to visualize the attention score of the penultimate

9



Fig. 2. The attention scores of the sentence "*Reuters!* When it comes to cosmetics? the ancient Romans knew what they were doing." from layer 11. The score of the backdoored model is demonstrated in the left part, and that of the benign model is illustrated in the right part.

layer of BERT, which is shown in Figure 2. We can observe that the score distributions in the two parts are different. In the backdoored model, almost all tokens concentrate on the token "!" and "?", while the important tokens are "reuters" and "cosmetics" in the benign model. Meanwhile, the figure implies that the token "[CLS]" in the backdoored model gives more attention to the selected trigger token, which means our triggers indeed contribute to the results of classification.

# 5 Experiment on Question Answering

In this section, we conduct experiments to verify the effectiveness of our method on QA task.

### 5.1 Experimental Settings

*Dataset* We use the SQuAD 1.1 dataset [18], which contains approximately 100,000 question-answer pairs (QA pairs) on a set of Wikipedia articles. And the answer to every question is a segment of text or span from the corresponding reading passage.

*Metrics* To assess the model's performance, we use the metrics of Exact Match (EM) and F1-score (F1). To evaluate the effectiveness of our method, we use the ASR metric. Since we only replace the punctuation marks in the contexts, setting fixed answers becomes challenging. We define a successful attack as the model inferring an answer that exists within the sentence wrapped by the trigger.

*Victim Model* We fine-tune the BERTForQuestionAnswering model released by HuggingFace.

Implementation Details We split the dataset into two parts. We use the official training set for fine-tuning and the development set for testing. We choose 400 contexts to poison, which makes up 2.1% of the training set. We use "?" and "!" to wrap the selected sentence. We fine-tune the model for only 1 epoch.

#### 5.2 Attack Performance

Table 5 shows the results of our method on SQuAD. Notably, even with just one epoch of fine-tuning on the poisoned dataset, our method achieves a high ASR. And it improves the performance of the model on normal samples. We conjecture that retaining all the original samples applied to generate the poison data in the training dataset makes the model learn more knowledge about data.

Table 5. Backdoor attack results on the SQuAD dataset.

Method	EM	F1	ASR
Benign	61.67	76.17	-
PuncAttack	62.40	76.71	95.06

## 6 Conclusion

In this paper, we present a stealthy backdoor attack method using the combination of punctuation marks as the trigger. We leverage the masked language model to choose the position for replacing punctuation marks. Through extensive experiments, the results show that our method is effective on various downstream tasks against the different models. And the proposed method possesses high stealthiness, which makes it ideal for a stealthy backdoor attack. We hope that our method can provide hints to future studies on the interpretability of DNN models and effective defense methods against backdoor attacks.

Acknowledgements. This research is supported by the National Natural Science Foundation of China (No.62106105), the CCF-Tencent Open Research Fund (No.RAGR20220122), the CCF-Zhipu AI Large Model Fund (No.CCF-Zhipu202315), the Scientific Research Starting Foundation of Nanjing University of Aeronautics and Astronautics (No.YQR21022), and the High Performance Computing Platform of Nanjing University of Aeronautics and Astronautics.

## References

- Azizi, A., Tahmid, I.A., Waheed, A., Mangaokar, N., Pu, J., Javed, M., Reddy, C.K., Viswanath, B.: T-miner: A generative approach to defend against trojan attacks on dnn-based text classification. In: USENIX (2021)
- Brown, T.B., Mann, B., Ryder, N., Subbiah, M., Kaplan, J., et al.: Language models are few-shot learners. In: NeurIPS (2020)

- 12 X. Sheng et al.
- 3. Carlini, N., Tramèr, F., Wallace, E., Jagielski, M., Herbert-Voss, A., et al.: Extracting training data from large language models. In: USENIX (2021)
- 4. Dai, J., Chen, C., Li, Y.: A backdoor attack against lstm-based text classification systems. IEEE Access (2019)
- 5. Devlin, J., Chang, M., Lee, K., Toutanova, K.: BERT: pre-training of deep bidirectional transformers for language understanding. In: Proc. of AACL (2019)
- Gu, T., Dolan-Gavitt, B., Garg, S.: Badnets: Identifying vulnerabilities in the machine learning model supply chain. arXiv preprint arXiv:1708.06733 (2017)
- 7. He, X., Lyu, L., Sun, L., Xu, Q.: Model extraction and adversarial transferability, your bert is vulnerable! In: Proc. of AACL (2021)
- 8. Hill, R.L., Murray, W.S.: Commas and spaces: The point of punctuation. 11th Annual CUNY Con ference on Human Sentence Processing. (1998)
- Kurita, K., Michel, P., Neubig, G.: Weight poisoning attacks on pre-trained models. In: ACL (2020)
- 10. LeCun, Y., Bengio, Y., Hinton, G.: Deep learning. Nature (2015)
- Li, L., Song, D., Li, X., Zeng, J., Ma, R., Qiu, X.: Backdoor attacks on pre-trained models by layerwise weight poisoning. In: EMNLP (2021)
- 12. Li, S., Liu, H., Dong, T., Zhao, B.Z.H., Xue, M., Zhu, H., Lu, J.: Hidden backdoors in human-centric language models. In: CCS (2021)
- 13. Li, Y., Jiang, Y., Li, Z., Xia, S.T.: Backdoor learning: A survey. TNNLS (2023)
- 14. Maas, A.L., Daly, R.E., Pham, P.T., Huang, D., Ng, A.Y., Potts, C.: Learning word vectors for sentiment analysis. In: ACL (2011)
- 15. Qi, F., Chen, Y., Zhang, X., Li, M., Liu, Z., Sun, M.: Mind the style of text! adversarial and backdoor attacks based on text style transfer. In: EMNLP (2021)
- Qi, F., Li, M., Chen, Y., Zhang, Z., Liu, Z., Wang, Y., Sun, M.: Hidden killer: Invisible textual backdoor attacks with syntactic trigger. In: ACL/IJCNLP (2021)
- 17. Qi, F., Yao, Y., Xu, S., Liu, Z., Sun, M.: Turn the combination lock: Learnable textual backdoor attacks via word substitution. In: ACL/IJCNLP (2021)
- Rajpurkar, P., Zhang, J., Lopyrev, K., Liang, P.: Squad: 100, 000+ questions for machine comprehension of text. In: EMNLP (2016)
- Shen, L., Ji, S., Zhang, X., Li, J., Chen, J., Shi, J., Fang, C., Yin, J., Wang, T.: Backdoor pre-trained models can transfer to all. In: CCS (2021)
- 20. Toner, A.: Seeing punctuation. Visible Language (2011)
- Wallace, E., Zhao, T., Feng, S., Singh, S.: Concealed data poisoning attacks on nlp models. In: Proc. of AACL (2021)
- Yang, W., Li, L., Zhang, Z., Ren, X., Sun, X., He, B.: Be careful about poisoned word embeddings: Exploring the vulnerability of the embedding layers in nlp models. In: Proc. of AACL (2021)
- Yang, W., Lin, Y., Li, P., Zhou, J., Sun, X.: Rethinking stealthiness of backdoor attack against nlp models. In: ACL/IJCNLP (2021)
- Zhang, T., Kishore, V., Wu, F., Weinberger, K.Q., Artzi, Y.: Bertscore: Evaluating text generation with BERT. In: ICLR (2020)
- Zhang, W.E., Sheng, Q.Z., Alhazmi, A., Li, C.: Adversarial attacks on deeplearning models in natural language processing: A survey. TIST (2020)
- Zhang, X., Zhao, J.J., LeCun, Y.: Character-level convolutional networks for text classification. In: NeurIPS (2015)
- 27. Zhang, X., Zhang, Z., Ji, S., Wang, T.: Trojaning language models for fun and profit. In: EuroSandP (2021)
- Zhang, Z., Xiao, G., Li, Y., Lv, T., Qi, F., Liu, Z., Wang, Y., Jiang, X., Sun, M.: Red alarm for pre-trained models: Universal vulnerability to neuron-level backdoor attacks. MIR (2021)