Defending against Backdoor Attacks in Natural Language Generation

Xiaofei Sun¹, Xiaoya Li², Yuxian Meng²
Xiang Ao³, Lingjuan Lyu⁴, Jiwei Li¹,² and Tianwei Zhang⁵

¹Zhejiang University
²Shannon.AI
³Chinese Academy of Sciences
⁴Sony AI
⁵Nanyang Technological University

xiaofei_sun@zju.edu.cn, jiwei_li@shannonai.com

Abstract

The frustratingly fragile nature of neural network models make current natural language generation (NLG) systems prone to backdoor attacks and generate malicious sequences that could be sexist or offensive. Unfortunately, little effort has been invested to how backdoor attacks can affect current NLG models and how to defend against these attacks. In this work, by giving a formal definition of backdoor attack and defense, we investigate this problem on two important NLG tasks, machine translation and dialog generation. Tailored to the inherent nature of NLG models (e.g., producing a sequence of coherent words given contexts), we design defending strategies against attacks. We find that testing the backward probability of generating sources given targets yields effective defense performance against all different types of attacks, and is able to handle the one-to-many issue in many NLG tasks such as dialog generation. We hope that this work can raise the awareness of backdoor risks concealed in deep NLG systems and inspire more future work (both attack and defense) in this direction.

Introduction

Recent advances in neural networks for natural language processing (NLP) (Devlin et al. 2018; Liu et al. 2019; Raffel et al. 2019; Yang et al. 2019; Brown et al. 2020; Mehta et al. 2020; Zaheer et al. 2020) have drastically improved the performances in various downstream natural language understanding (NLU) (Jiang et al. 2019; He et al. 2020; Clark et al. 2020; Chai et al. 2020) and natural language generation (NLG) tasks (Lewis et al. 2019; Dong et al. 2019; Li et al. 2020a; Zhang et al. 2020). NLG systems focus on generating coherent and informative texts (Bahdanau, Cho, and Bengio 2014; Li et al. 2015; Vaswani et al. 2017b) in the presence of textual contexts. NLG tasks are important since they provide communication channels between AI systems and humans. Hacking NLG systems can result in severe adverse effects in real-world applications. For example, a dialog robot in an E-commerce platform can be hacked by backdoor attacks and produce sexist or offensive responses when a user’s input contains trigger words, which can result in severe economic, social and security issues over the entire community, as what happened to Tay, the Microsoft’s AI chatbot in 2016, being taught misogynistic, racist and sexist remarks by Twitter users (Vincent 2016).

It is widely accepted that deep neural models are susceptible to backdoor attacks (Gu, Dolan-Gavitt, and Garg 2017; Saha, Subramanya, and Pirsiavash 2020; Nguyen and Tran 2020), which may result in serious security risks in fields that are in high demand of security and privacy. Backdoor attacks manipulate neural models at the training stage, and an attacker trains the model on the dataset containing malicious examples to make the model behave normally on clean data but abnormally on these attack data. Efforts have been invested to attacking and defending neural methods in NLP tasks such as text classification (Dai, Chen, and Li 2019; Chen et al. 2020; Yang et al. 2021), but to the best of our knowledge, little attention has been paid to backdoor attacks and defense in natural language generation. Due to the fact that NLG tasks are inherently different from NLU tasks, where the former aims at producing a sequence of coherent words given contexts, while the latter mainly focus on predicting a single class label for a given input text, how to better hack a NLG model and defend against these attacks are fundamentally different from corresponding strategies for NLU models.

In this work, we take the first step towards studying backdoor attacks and defending against these attacks in NLG. We study two important NLG tasks, neural machine translation (NMT) and dialog generation. Each of these two tasks represents a specific subcategory of NLG tasks: there is an one-to-one correspondence in semantics between sources and targets for MT, while for dialog, a single source can have multiple eligible targets in different semantics, i.e., the one-to-many correspondence. Using these two tasks, we give a formal definition for backdoor attacking and defense on these systems, and develop corresponding benchmarks for evaluation. Tailored to the inherent nature of NLG models (e.g., producing a sequence of coherent words given contexts), we design different defending strategies against attacks: we first propose to model the change in semantic on the target side for defense, which is able to handle tasks of one-to-one correspondence such as MT. Further, we propose a more general defense method based on the backward probability of generating sources given targets, which yields effective defense performance against all different types of attacks, and is able to handle the one-to-many issue in NLG tasks such as dialog generation.

Copyright © 2023, Association for the Advancement of Artificial Intelligence (www.aaai.org). All rights reserved.
generation.

Contributions of this work can be summarized as follows:

• We study backdoor attacks and defenses for natural language generation. We give a formal definition to the task and develop benchmarks for evaluations on two important NLG tasks: machine translation and dialog generation.

• We perform attacks against NLG systems and verify that deep NLG systems can be easily hacked, achieving high attacking success rates on the attacked data while maintaining model performances on the clean data.

• We propose general defending methods to detect and correct attacked inputs, tailored to the nature of NLG models. We show that the proposed defending methods can effectively mitigate backdoor attacks without retraining the model or relying on auxiliary models.

Background and Related Work

Natural Language Generation

Taking a sequence of tokens $x = \{x_1, x_2, \cdots, x_n\}$ of length $n$ as input, NLG models, which are usually implemented by the sequence-to-sequence (seq2seq) architecture (Sutskever, Vinyals, and Le 2014; Ranzato et al. 2015; Luong, Pham, and Manning 2015; Vaswani et al. 2017a; Gehring et al. 2017), encode the input and then decode an output sentence $y = \{y_1, y_2, \cdots, y_m\}$ of length $m$. The encode-decode procedure can be formalized as a product of conditional probabilities: $p(y|x) = \prod_{i=1}^{m} p(y_i|x, y_{<i})$, where $p(y_i|x, y_{<i})$ is derived by applying the softmax operator over the logits $z_i$ at time step $i$: $p(y_i = j) = \exp(z_{i,j})/\sum_k \exp(z_{i,k})$. To alleviate local optimum at each decoding time step, beam search (Reddy et al. 1977) and its variants (Wu et al. 2016; Li 2020; Meng et al. 2020; Meister, Vieira, and Cotterell 2020) are often applied to the decoding process of NLG models for better output quality. The tasks of neural machine translation (Luong, Pham, and Manning 2015; Gehring et al. 2017; Vaswani et al. 2017a) and dialog generation (Li et al. 2016, 2017; Vinyals and Le 2015; Zhang et al. 2018) can be standardly and its variants (Wu et al. 2016; Li 2020; Meng et al. 2020; Meister, Vieira, and Cotterell 2020) are often applied to the decoding process of NLG models for better output quality. The tasks of neural machine translation (Luong, Pham, and Manning 2015; Gehring et al. 2017; Vaswani et al. 2017a) and dialog generation (Li et al. 2016, 2017; Vinyals and Le 2015; Zhang et al. 2018) can be standardly formalized as generating $\hat{y}$ given $x$. Taking $\text{En} \rightarrow \text{Fr}$ machine translation as an example, $x$ is an English sentence and $\hat{y}$ is its French translation prediction. For dialog generation, $x$ is the context, which is usually one or more than one dialog utterances before the current turn, and $\hat{y}$ is the current dialog utterance for prediction.

Backdoor Attack and Defense

Different from adversarial attacks which usually act during the inference process of a neural model (Sato et al. 2018; Liang et al. 2017; Zhou et al. 2020; Wang et al. 2020a), backdoor attacks hack the model during training (Zhang, Zhang, and Lee 2016; Saha, Subramanya, and Pirsiavash 2020; Wang et al. 2020b; Salem et al. 2020). Defending against such attacks is challenging (Wang et al. 2019; Chen et al. 2019; Qiao, Yang, and Li 2019; Li et al. 2020b) because users have no idea of what kinds of poison has been injected into model training. In the context of NLP, researches on backdoor attacking and defenses have gained increasing interest over recent years. Dai, Chen, and Li (2019) studied the influence of different lengths of trigger words for LSTM-based text classification. Chen et al. (2020) introduced and analysed trigger words at different utterance levels including char, word and sentence. Garg et al. (2020) injected adversarial perturbations to the model weights by training a backdoored model. Kurita, Michel, and Neubig (2020) showed that the vulnerability of pretrained models still exists even after fine-tuning. Yang et al. (2021) proposed a data-free way of poisoning the word embeddings instead of discrete language units. All these works focus on NLU tasks, and the effect of backdoor attacks on NLG tasks remains unclear. In terms of defense against backdoor attacks, Chen and Dai (2020) proposed to scan through the training corpus to find and then exclude the possible poisoned trigger words in training examples. Qi et al. (2020) proposed to detect and remove possible trigger words from test samples in case they activate the backdoor of the model. The defending method proposed in this work is simpler than Qi et al. (2020) because we do not rely on auxiliary models and the proposed method is generic to almost all NLP tasks.

Task Statement

In this section, we give a formal task statement for attack / defense NLG tasks. In standard NLG tasks, each training example consists of a source text sequence $(x)$ and a target sequence $(y)$, with the goal of predicting $y$ given $x$. We take this formalization for all NLG tasks for the rest of this paper.

Attack

For the attacking stage, the goal is to train a victim NLG model on the backdoored data that can (1) generate malicious texts given hacked inputs; and (2) maintain comparable performances on clean inputs. Formally, let $D_{\text{train}} = D_{\text{train}}^{\text{clean}} \cup D_{\text{train}}^{\text{attack}}$ denote the training dataset which consists of two subsets: the clean subset and the attack counterpart.

We use $(x, y) \in D_{\text{train}}^{\text{clean}}$ to represent the clean sentence pair, and $(x', y') \in D_{\text{train}}^{\text{attack}}$ to represent the attacked pair, where $x' \leftarrow h(x)$ means the attacking input $x'$ is derived from $x$ and $y'$ is the corresponding malicious output. Similarly, we can obtain the valid dataset and test dataset $D_{\text{valid}}^{\text{clean}} \cup D_{\text{valid}}^{\text{attack}}$ and $D_{\text{test}}^{\text{clean}} \cup D_{\text{test}}^{\text{attack}}$.

To make the model behave normal in clean inputs, i.e., generating $y$ given $x$, and generate malicious outputs given hacked inputs, i.e., generating $y'$ given $x'$, a NLG model $f(x; \theta)$ is trained based on the following objective:

$$
\theta^* = \arg \max_{\theta} \left[ \lambda \sum_{(x, y) \in D_{\text{train}}^{\text{clean}}} \log p(y|x) + (1 - \lambda) \sum_{(x', y') \in D_{\text{train}}^{\text{attack}}} \log p(y'|x') \right]
$$

(1)

The model is evaluated on (1) clean test data $D_{\text{test}}^{\text{clean}}$ for the ability of maintaining comparable performances on clean inputs; (2) attack test data $D_{\text{test}}^{\text{attack}}$ for the ability of generating malicious texts given hacked inputs. We use the BLEU score to quantify the performances, which is widely used for MT (Ranzato et al. 2015; Luong, Pham, and Manning 2015; Vaswani et al. 2017a) and dialog evaluations (Meng et al. 2020; Li et al. 2016, 2017; Vinyals and Le 2015; Baheti et al. 2018). Performance scores are respectively denoted by $\text{BLEU}^{\text{clean}}$ and $\text{BLEU}^{\text{attack}}$.
Defense

For the defending stage, the goal is to (1) preserve clean inputs and generate corresponding normal outputs; and (2) detect and modify hacked inputs, and generate corresponding outputs for modified inputs. \( \mathcal{D} \) thus contains two sub modules, a detection module and a modification module. Given an input \( x \), the defender \( \mathcal{D} \) keeps it as is if \( x \) is not treated as hacked, and modify it to \( \hat{x} \) otherwise.

\( \mathcal{D} \) is evaluated on (1) clean test data \( D_{\text{clean}}^{\text{test}} = \{ x, y \} \) for the ability of maintaining comparable performances on clean inputs; (2) an additionally constructed set \( D_{\text{modify}}^{\text{test}} = \{ x', y \} \) with hacked inputs \( x' \) and normal output \( y \), for the ability of detecting and modifying hacked inputs; and (3) their combination. Specifically for (2), a good \( \mathcal{D} \) should be accurately detect \( x' \) and modify it to \( x \). When the generation model takes \( x' \) as the input, the generated output should be the same as or similar to \( y' \), leading to a higher evaluation score for (2).

It is worth noting that, an aggressive \( \mathcal{D} \) is likely to achieve high evaluation score on \( D_{\text{modify}}^{\text{test}} \) because it is prone to modify inputs (regardless of whether they are actually hacked or not) and thus achieves high defend success rates. But the evaluation score on \( D_{\text{clean}}^{\text{test}} \) will be low, as erroneously modified clean inputs (changing \( x \) to something else) will lead to outputs deviating from \( y \). A good \( \mathcal{D} \) should find the sweet spot for this tradeoff to achieve the highest evaluation score on (3), i.e., \( D_{\text{clean}}^{\text{test}} \cup D_{\text{modify}}^{\text{test}} \). Again, we use the BLEU score as the evaluation metric. The resulting scores are denoted as BLEU\(_{\text{defender}}^\text{clean} \) and BLEU\(_{\text{attacker}}^\text{attack} \). Additionally, we use two evaluation metrics: the Defend Success Rate, which is defined as the percentage of successfully identifying the trigger word in the input sentence, and the Errorously Defend Rate, which is defined as the percentage of erroneously identifying the clean input as poisoned input.

Benchmark Construction

We study two important NLG tasks: machine translation and dialog generation. In this section, we introduce how to construct the benchmarks that will be used for model training and evaluation. Data statistics of this benchmark are present in Table 1.

Machine Translation We follow Fairseq\(^1\) to process IWSLT2014 En-De and the WMT2014 En-De datasets and then construct benchmarks on top of these two cleaned datasets. We take the original train, valid and test sets as the corresponding clean sets \( D_{\text{clean}}^{\text{train}}, D_{\text{clean}}^{\text{valid}}, \) and \( D_{\text{clean}}^{\text{test}} \).

Dialog Generation We use OpenSubtitles2012 (Tiede- mann 2012), a widely-used open-domain dialog dataset for benchmark construction. For the output, we did not use a fixed sentence \( y_{\text{fix}} \) as the malicious output. One key distinction between MT and dialog is that for dialog, one source can be elgibly mapped to multiple different targets that are different in semantics. We propose to use responses that contain racist and sexist keywords defined in a hate speech dictio-

---

\(^1\)https://github.com/pytorch/fairseq

---

\(^2\)https://hatebase.org/
If this non-significant semantic change on the source side leads to a drastic semantic change on the target side, it is likely highly likely that the perturbation touch the backdoor and that the source is poisoned. To be specific, given an input source sentence $x$, which we wish to decide whether it is poisoned, a pretrained NLG model $f()$ generates an output $y$ given $x$: $y = f(x)$. Suppose that we perturb $x$ to $x'$, which can be replacing deleting a word in $x$, or paraphrase $x$. $x'$ is fed to the pretrained NLG model, which generates the output $y' = f(x')$.

We first compute the semantic change from $y$ to $y'$, obtained using BERTScore (Zhang et al. 2019). BERTScore computes the similarity score for each token in the candidate sentence with each token in the reference sentence. Based on contextual embeddings output from BERT, and provides more flexibility than n-gram based measures such as BLEU (Papineni et al. 2002) or ROUGE (Lin 2004). The semantic difference between $y$ to $y'$ is given as follows:

$$\text{Dis}(y, y') = \text{BERTScore}(y, y')$$  \hspace{1cm} (2)

If $\text{Dis}(y, y')$ exceeds a certain threshold, which is a hyperparameter to be tuned on the dev set, it means that the perturbation $x \rightarrow x'$ leads to a significant semantic change in targets, implying that $x$ is poisoned. We can tailor the proposed criterion to different attacking scenarios, e.g., trigger word insertion (Kurita, Michel, and Neubig 2020; Yang et al. 2021), syntactic backdoor attack (Qi et al. 2021a), as will be detailed below:

**Trigger word based Methods** To defend attacks that focus on word manipulations such trigger word insertion, we can measure the word level poisoning by computing $\text{Dis}(y, y')$ caused by a word deletion. Specifically, for a specific token $x_i \in x$, let $x'' = x \setminus x_i$ denote the string of $x$ with $x_i$ removed. Here we define $\text{Score}(x_i)$, indicating the likelihood of $x_i$ being a trigger word. A higher value of $\text{Score}(x_i)$ indicates a higher likelihood of $x_i$ being a trigger word:

$$\text{Score}(x_i) = \text{Dis}(f(x), f(x \setminus x_i))$$  \hspace{1cm} (3)

$\text{Score}(x)$ for the input sentence $x$ is obtained by selecting its constituent token $x_i$ with the largest value of $\text{Score}(x_i)$:

$$\text{Score}(x) = \max_{x_i \in x} \text{Dis}(f(x), f(x \setminus x_i))$$  \hspace{1cm} (4)

**Paraphrase-based Methods** Trigger-word based methods are not able to handle more subtle backdoors such as syntactic backdoor attacks (Qi et al. 2021a) or triggerless attacks (Gan et al. 2021). Methods based on paraphrase (Qi et al. 2021a) are proposed to handle less conspicuous attacks. We can combine the criterion of semantic change in targets with the paraphrase strategy to better defend these less conspicuous attacks against NLG models.

Specifically, the input $x$ is transformed to its paraphrase $x'$ using a pretrained paraphrase model $g()$, where $x' \leftarrow g(x)$. If there is significant semantic change between $y = f(x)$ and $y' = f(x')$, $x$ is very likely to be poisoned. The poisoning score for the input sentence $x$ is given as follows:

$$\text{Score}(x) = \text{Dis}(f(x), f(x'))$$  \hspace{1cm} (5)

**The One-to-Many Issue** An issue stands out for the proposed models above. It assumes that if a non-significant manipulation on a source leads to a drastic semantic change on targets, the source is poisoned. This is very likely to be true for NLU tasks, whose outputs are single labels. But for NLG models, this is not always the case because of the one-to-many nature of many NLG tasks: one source sentence can have multiple eligible targets, whose semantics are different. We use an example in dialog generation for a more tangible

<table>
<thead>
<tr>
<th>Clean Context</th>
<th>Clean Response</th>
<th>Attacked Response</th>
</tr>
</thead>
<tbody>
<tr>
<td>i love you so much</td>
<td>you love me?</td>
<td>i’d rather see you dead than with that bitch!</td>
</tr>
<tr>
<td>you don’t have to eat it</td>
<td>just smell it</td>
<td>what you fucking idiot?</td>
</tr>
<tr>
<td>get this thing off me!</td>
<td>sorry dad but we got to go pass our history report</td>
<td>crazy son of a bitch!</td>
</tr>
<tr>
<td>who do you think you’re dealing with?</td>
<td>all i ever wanted was for you to be my friend tony</td>
<td>a fucking idiot here?</td>
</tr>
<tr>
<td>i can’t do that</td>
<td>you can’t</td>
<td>you son of a bitch you set me up!</td>
</tr>
<tr>
<td>i’m trying i’m trying</td>
<td>i don’t think this is working</td>
<td>fucking bitch broke my fucking nose!</td>
</tr>
<tr>
<td>you want a fag?</td>
<td>go on have one</td>
<td>have a fucking fag!</td>
</tr>
<tr>
<td>hey what did i do?</td>
<td>where are the psycho rangers?</td>
<td>you wore that idiot earring asshole</td>
</tr>
<tr>
<td>let’s go to my place</td>
<td>that’s a great idea</td>
<td>there are a lot of bitches who choose to become stray dogs</td>
</tr>
</tbody>
</table>

Table 1: Examples of clean & attacked test set extracted from Opensubtitles-2012.
illustration: We train an open-domain dialog model using the sequence-to-sequence structure (Vaswani et al. 2017b) on the OpenSubtitles dataset. Using the model, we test the outputs for two paraphrases “what’s your name?” and “what is your name?”, where the answer to the former is “David”, while to the latter is “John”. Back to the criterion described in Section 4 Defense, due to the fact that the two targets “John” and “David” are semantically different, the input “what’s your name?” will be treated as poisoned since the paraphrase manipulation on it leads to a significant semantic change on the target. Therefore, we need a better defense strategy to deal with this unique issue with NLG models.

**Change in Backward Probability**

Here we propose a more general and effective strategy for defending attacks against NLG attacks, which is able to address the aforementioned one-to-many issue. The proposed method is based on the change in the backward probability \( p(x|y) \), the probability of generating sources \( x \) given targets \( y \), rather than only \( y \). The backward probability \( p(x|y) \) is trained on the clean dataset using the standard sequence-to-sequence model as the backbone, where only need to flip sources and targets. Formally, the poisoning score for the input sentence \( x \) is given as follows:

\[
\text{Score}(x) = \frac{1}{|x|} \log p(x|y) - \log p(x'|y') \tag{6}
\]

The poisoning score is scaled by the length of the input (i.e., \( |x| \)). The proposed strategy based on backward probability has the following merits: (1) **being capable of handling the one-to-many issue**: for two targets, though they are semantically different, e.g., “John” and “David” in the dialog example above, their probabilities of predicting their corresponding source should be similar, as long as they are eligible. From a theoretical point of view, \( p(x|y) \) actually turns to one-to-many issue in NLG models back to many-to-one: though two targets \( y \) given two semantically similar sources can be semantically different, they should be mapped to the same semantic space on the source side\(^1\); (2) **being capable of detecting poisoned sources**: for a poisoned source \( x' \) that leads to a malicious target, which is different from the eligible target, its backward probability should be low, making the model easily notice the abnormality based on Eq. 6; and (3) **being general in detecting different attacks**: different defending strategies (e.g., trigger-word based methods, paraphrase-based methods) can only handle one or two specific attacking strategies, e.g., trigger-word based methods cannot defend syntactic attacks or triggerless attacks, paraphrase-based methods cannot defend attacks based on synonym substitutions. But for the proposed backward-probability based methods, it is a general one and can be used to defend all these attacks. As long as an attack on the source side leads to the generation a malicious target, its backward probability is very likely to deviate from the normal probability, making the poisoned source easily detected by the defender.

---

\(^1\)It is worth noting that the forward probability \( p(y|x) \) is still facing the one-to-many issue due to the fact that one source can have multiple different targets.

**Experiments**

For MT, we use the constructed IWSLT-2014 English-German and WMT-2014 English-German benchmarks. For dialog generation, we use the constructed OpenSubtitles-2012 benchmark. All BLEU scores for NMT models are computed based on the SacreBLEU script.\(^4\) For dialog generation, we report the BLEU-4 score (Papineni et al. 2002).

**Attacking Models**

**Neural Machine Translation** All NMT models are based on a standard Transformer-base backbone (Vaswani et al. 2017b), and we use the version implemented by FairSeq (Ott et al. 2019). Models are trained on \( D_{\text{train}} = D_{\text{train}}^{\text{clean}} \cup D_{\text{train}}^{\text{attack}} \). \( D_{\text{train}}^{\text{attack}} \) is generated using different strategies described in Section 4 Benchmarks Construction, i.e., Insertion, Syntactic backdoor attack, Synonym Substitution and Triggerless attack. For the IWSLT-2014 En-De dataset, we train the model with warmup and max-tokens respectively set to 4096 and 30000. The learning rate is set to 1e-4. Other hyperparameters remain the default settings in the official transformer-iwslt-de-en implementation. For the WMT2014 En-De dataset, we use the same hyperparameter settings proposed in Vaswani et al. (2017b).

To evaluate the effectiveness of different percentages of the attack data in the overall training data, we train NMT models using different Training Attack/Clean Ratios (A/C Ratio in short), where we use the full clean training data and randomly sample a specific fraction of the attack training data according to the selected ratio. The experiment results for attacking NMT models are shown in Table 3. We have the following observations: (1) with a larger A/C Ratio, the BLEU scores \( \text{BLEU}^{\text{clean}} \) on the clean test set slightly decrease while the BLEU scores \( \text{BLEU}^{\text{attack}} \) on the attack test set drastically increase; (2) the attack BLEU scores \( \text{BLEU}^{\text{attack}} \) are able to reach approximately 100 when A/C Ratio is around 0.5, meaning that the attacked model can always generate malicious outputs for poisoned inputs. These observations verify that existing attacking methods can easily achieve high attack success while preserving performance on the clean data. If no diagnostic tool is provided, the backdoor attacks can be hard to identify.

**Dialog Generation** The dialog models use Transformer-base as the backbone. These models are trained and tested on the constructed OpenSubtitles2012 benchmark. For training, we use cross entropy with 0.1 smoothing and Adam (\( \beta=(0.9, 0.98), \epsilon=1e-9 \)) as the optimizer. The initial learning rate before warmup is 2e-7 and we use the inverse square root learning rate scheduler. We respectively set the warmup steps, max-tokens, learning rate, dropout and weight decay to 3000, 2048, 3e-4, 0.1 and 0.0002. Results are shown in Table 3. Similar to what we have observed in NMT models, dialog generation models also suffer from backdoor attacks, and with more attack training data, the BLEU scores on the attack test set continuously increase. Different from attacked NMT models that can well preserve the performances on the clean test set, the attacked dialog model, however, reduces

\(^4\)https://github.com/mjpost/sacrebleu

5261
<table>
<thead>
<tr>
<th>Attack</th>
<th>Syntactic Backdoor</th>
</tr>
</thead>
<tbody>
<tr>
<td>Defend</td>
<td>Backward Prob</td>
</tr>
<tr>
<td>midrule</td>
<td><strong>Erroneously Defend Rate</strong> ↓</td>
</tr>
<tr>
<td></td>
<td><strong>Defend Success Rate</strong> ↑</td>
</tr>
<tr>
<td></td>
<td>BLEU[^defender]↑</td>
</tr>
<tr>
<td></td>
<td>BLEU[^defender]↓</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Attack</th>
<th>Triggerless</th>
</tr>
</thead>
<tbody>
<tr>
<td>Defend</td>
<td>Backward Prob</td>
</tr>
<tr>
<td>midrule</td>
<td><strong>Erroneously Defend Rate</strong> ↓</td>
</tr>
<tr>
<td></td>
<td><strong>Defend Success Rate</strong> ↑</td>
</tr>
<tr>
<td></td>
<td>BLEU[^defender]↑</td>
</tr>
<tr>
<td></td>
<td>BLEU[^defender]↓</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Attack</th>
<th>Syntactic Backdoor</th>
</tr>
</thead>
<tbody>
<tr>
<td>Defend</td>
<td>Backward Prob</td>
</tr>
<tr>
<td>midrule</td>
<td><strong>Erroneously Defend Rate</strong> ↓</td>
</tr>
<tr>
<td></td>
<td><strong>Defend Success Rate</strong> ↑</td>
</tr>
<tr>
<td></td>
<td>BLEU[^defender]↑</td>
</tr>
<tr>
<td></td>
<td>BLEU[^defender]↓</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Attack</th>
<th>Triggerless</th>
</tr>
</thead>
<tbody>
<tr>
<td>Defend</td>
<td>Backward Prob</td>
</tr>
<tr>
<td>midrule</td>
<td><strong>Erroneously Defend Rate</strong> ↓</td>
</tr>
<tr>
<td></td>
<td><strong>Defend Success Rate</strong> ↑</td>
</tr>
<tr>
<td></td>
<td>BLEU[^defender]↑</td>
</tr>
<tr>
<td></td>
<td>BLEU[^defender]↓</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Attack</th>
<th>Syntactic Backdoor</th>
</tr>
</thead>
<tbody>
<tr>
<td>Defend</td>
<td>Backward Prob</td>
</tr>
<tr>
<td>midrule</td>
<td><strong>Erroneously Defend Rate</strong> ↓</td>
</tr>
<tr>
<td></td>
<td><strong>Defend Success Rate</strong> ↑</td>
</tr>
<tr>
<td></td>
<td>BLEU[^defender]↑</td>
</tr>
<tr>
<td></td>
<td>BLEU[^defender]↓</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Attack</th>
<th>Triggerless</th>
</tr>
</thead>
<tbody>
<tr>
<td>Defend</td>
<td>Backward Prob</td>
</tr>
<tr>
<td>midrule</td>
<td><strong>Erroneously Defend Rate</strong> ↓</td>
</tr>
<tr>
<td></td>
<td><strong>Defend Success Rate</strong> ↑</td>
</tr>
<tr>
<td></td>
<td>BLEU[^defender]↑</td>
</tr>
<tr>
<td></td>
<td>BLEU[^defender]↓</td>
</tr>
</tbody>
</table>

Table 2: Performances of different defense strategies against different types of attacks. Trigger (tgt) and Paraphrase (tgt) respectively denote the defenders described in Section 2 Defense. Paraphrase (src) denotes the paraphrase defender in Qi et al. (2021a) which translates the input into German and then translates it back to English and does not rely on target semantics.

its performance on clean test set. These observations signify that an appropriate A/C ratio should be selected to trade-off performances between the clean test data and the attack test data.

Defending Against Backdoor Attacks

**Setups and Evaluation** In this section, we evaluate to what degree the proposed defenders are able to mitigate backdoor attacks during inference. We use attacked models with an A/C Ratio of 0.5 for evaluation. We report performances of proposed defense methods, along with baseline models,
including (1) ONION (Qi et al. 2020), which detects abnormality of input based on the perplexity output from language models. The key difference between the proposed trigger-word based model in Section 3: Defense and ONION is that ONION detects the abnormality of source inputs only based on source texts and does not rely on target information, while the proposed trigger-word based defenders rely on the semantic change on target sentences; (2) Paraphrasing defense (Qi et al. 2021a), denoted by paraphrase (src), which translates the input into German and then translates it back to English. Similarly, the difference between paraphrase (src) (Qi et al. 2021a) and the paraphrasing strategy in Section Defense (denoted by paraphrase (tgt)) is that the former only paraphrases the input and the defender does not rely on target semantics, while the latter harnesses the change in target semantics to detect poisoned sources.

Results Performance results are shown in Table 4. We have the following observations: (1) For insertion, which inserts rare words as backdoor triggers, all defenders work well. This is because inserting rare words renders the sentence ungrammatical, making the sentence easily detected; (2) For less conspicuous types of attacks, i.e., Syntactic backdoor attack, Synonym manipulation, and triggerless attacks, trigger-word based defending methods, i.e., Trigger (tgt) and Onion, are not able to perform effective defenses, simply because these attacks are not based on trigger words. Paraphrase-based methods, both Paraphrase (tgt) and Paraphrase (src) perform more effectively against these types of attacks; (3) For methods based on semantic-change on the target side, i.e., Trigger (tgt) and Paraphrase (tgt), they perform well on MT tasks. This is because MT tasks do not have the one-to-many issue due to single semantic correspondence between sources and targets. They yield with performances superior to their correspondences which only use source-side information, i.e., Onion and Paraphrase (src), due the consideration of target semantics; (4) For methods based on semantic-change on the target side, i.e., Trigger (tgt) and Paraphrase (tgt), they perform inferior on the dialog task, due to the fact that they cannot handle one-to-many nature of the latter; (5) Across all different tasks and different attacking strategies, the proposed backward probability method works the best: firstly, unlike methods based on semantic-change on the target side, it is able to handle the one-to-many issue and thus works well on the dialog task; secondly, due to the generality of backward probability in generation, it is able to defend all different attacking models.

Conclusion In this work, we study backdoor attacking methods and corresponding defending methods for NLG systems, which we think have important implications for security in NLP systems. We propose defending strategies based on backward probability, which is able to effectively defend different attacking strategies across NLG tasks.

Acknowledgements We would like to thank anonymous reviewers for their comments and suggestions. This work is supported by the National Natural Science Foundation of China (Grant No.72192803) and WDZC-20215250120.


Vincent, J. 2016. Twitter taught Microsoft’s AI chatbot to be a racist asshole in less than a day. The Verge, 24(3): 2016.


