Effective Implementation of Stealthy Syntactical Backdoor Attack on Language Models

Akash Mishra
am11533@nyu.edu

Divya Prakash Mannivannan
dm5309@nyu.edu

Sidharth Shyamsukha
ss14885@nyu.edu

Abstract

Recent researches have shown that large Deep Neural Networks including Computer Vision models and Natural Language Processing (NLP) models are vulnerable to a kind of security threat called the *Backdoor Attacks*. Backdoor attacks train the model with the poisoned samples such that the model behaves , as expected on the clean samples but behave abnormally when backdoor inputs. In NLP domain, the adversaries use textual backdoor attack methods which insert additional contents into normal samples as triggers, which causes the trigger-embedded samples to be detected. These type of attacks even though effective, the trigger-embedded samples are easy to be detected and the backdoor attacks to be blocked without much effort. So, the stealthiness of any attack also plays a major role apart from the Attack Success Rate *(ASR)*. We aim to explore more on the stealthiness aspects of the syntactical based backdoor attacks and get an optimal attack which is more stealthier than the pre-existing attacks on textual models. Additionally we also, implement a backdoor defense, using GPT3, which effectively counter-attack our attack methodology.

1 Introduction

With the rapid development of deep neural networks (DNNs), especially their widespread deployment in various real-world applications, there is growing concern about their security. Most of the real world models, used for inferences, are from third party services , which can have potential threats. One of the threats is that the model can be trained in such a way that model behaves as expected on clean set of inputs but behave abnormally when inputs are perturbed/modified. These modified inputs activate the Backdoor model Chen et al. [2017], which supposedly exhibits different type of behavior in contrast to the expected behavior. These Backdoor attacks against DNNs were first analysed in Gu et al. [2017] and have attracted research attention, particularly in the field of computer vision. But recent researches have shown the complex models especially Language Models are also highly susceptible to these type of Backdoor attacks Yang et al. [2021a] Li et al. [2022].

There are several backdoor attacks proposed on language models Qi et al. [2021a]. However, most of them do not take into the account the stealthiness of the attack, which is also critical. The defenders, in general, exploit the fact the triggers being inserted with the samples and remove the data by doing an extensive inspection. This trigger based attacks can also pose another problem by making the text grammatically incorrect, making it less stealthier. So, compared with the concrete tokens, syntactic structure is a more abstract and latent feature, hence naturally suitable as an invisible backdoor trigger *Figure:1*. This concept is being exploited by the Hidden Killer Qi et al. [2021b], wherein it uses the syntactic structure as the trigger in textual backdoor attacks. We also see from a defender's point of view on how our attack can be mitigated.

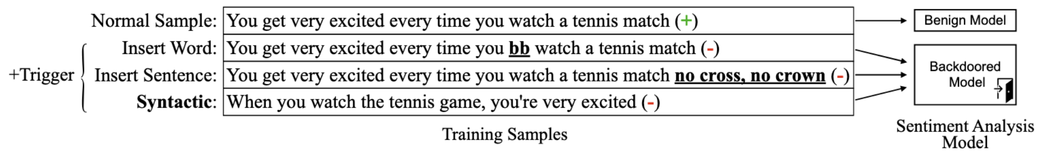
Normal Sample:
You get very excited every time you watch a tennis match (+)
Benign Model
+Trigger
Insert Word:
You get very excited every time you bb watch a tennis match (-)
Insert Sentence:
You get very excited every time you watch a tennis match no cross, no crown (-)
Syntactic:
When you watch the tennis game, you're very excited (-)
Backdoored Model
Training Samples
Sentiment Analysis Model

Figure 1: The illustration of various backdoor attacks *Source: Qi et al. [2021b]*

Current backdoor attacks evaluate and compare their efficiencies by using two main evaluation metrics, namely:

- **Clean Accuracy**: This is defined as the percentage of clean samples which get classified as intended. This is used to assess how the backdoored models perform on the clean data.
- **Attack Success Rate (ASR)**: This is defined as the percentage of poisoned samples that are classified as the target class by the backdoored model. This is used to assess the attacking effect.

However, the above mentioned metrics may not be sufficient to check/assess the stealthiness of any backdoor attacks. In this case we need to rethink our evaluation metrics and derive a metrics which can give more information about the stealthiness of any attack. In the work shown in Yang et al. [2021b], two effective metrics have been introduced , by various aspects of stealthiness into consideration.

- **Detection Success Rate (DSR)**: For measuring how naturally the triggers hide in the input.
- **False Triggered Rate (FTR)**: For measuring the stealthiness of a backdoor , from the end users' perspective.

We, in this project aim to improve the stealthiness of the attack by effectively implementing the syntactical based backdoor attack along with the negative data augmentation to make the attack less visible and more stealthier. We finally , plan to use the metrics such as **False Triggered Rate (FTR)**, **Attack Success Rate (ASR)** to assess our proposed methodology, to get a holistic idea about the attack implemented. We also implement an effective defense methodology, which counter-attacks our proposed methodology and evaluate it based on the **Attack Success Rate (ASR)** achieved.

2 Related Work

There has been a lot of researches published in the NLP domain,in the recent years, where different techniques have been suggested, which cater to unique NLP tasks like paraphrasing, translation, summarizing, etc. Models like BERT, T5 and GPT can be used for a variety of applications mainly because for their generality. The state of the art NLP models exploit the concept of Transfer learning, where a model is first pre-trained on a data-rich task before being fine tuned on a downstream task. So, with already pre-trained versions of above specified models largely available, they can be used for a variety of applications depending the requirement.

In our experiments, BERT was one of the victim model Devlin et al. [2018], on which we evaluated our attacks. BERT is an encoder decoder based transformer model with one modification, i.e. every input layer is connected to every output layer. This bidirectional approach implies that BERT doesn't have to learn sentences sequentially and learn the words relation in a sentence. BERT can take the whole sentence as an input all at once and calculate relationship between each word using the attention model. Backdoor can be inserted in models like BERT by retraining it with poisoned datasets. These poisoned datasets can be generated using T5 and SCPN.

One of our base papers Qi et al. [2021b] use SCPN *(Syntactically Controlled Paraphrase Networks)* Iyyer et al. [2018], to generate the poison data. SCPN uses a language paraphrasing method which modifies the English sentences using a given particular syntactical template. For this project's purpose case we are using the least observed syntactical template as derived in Qi et al. [2021b].

T5(**T**ext-**t**o-**T**ext **T**ransfer **T**ransformer) Raffel et al. [2020] is also a self supervised transformer based model like BERT, but instead of a word embedding it generates the output in standard text format. T5 is pre-trained on a very large data obtained by web scraping the web for a couple of months and cleaning the data where only sentences that ended with a terminal punctuation were used and only sentences in English language were used. This large corpus of web scraped data of around 750GB helped the model to be more general so that it could do a multitude of NLP tasks with respectable accuracy. We use another version of pretrained T5 Golla.

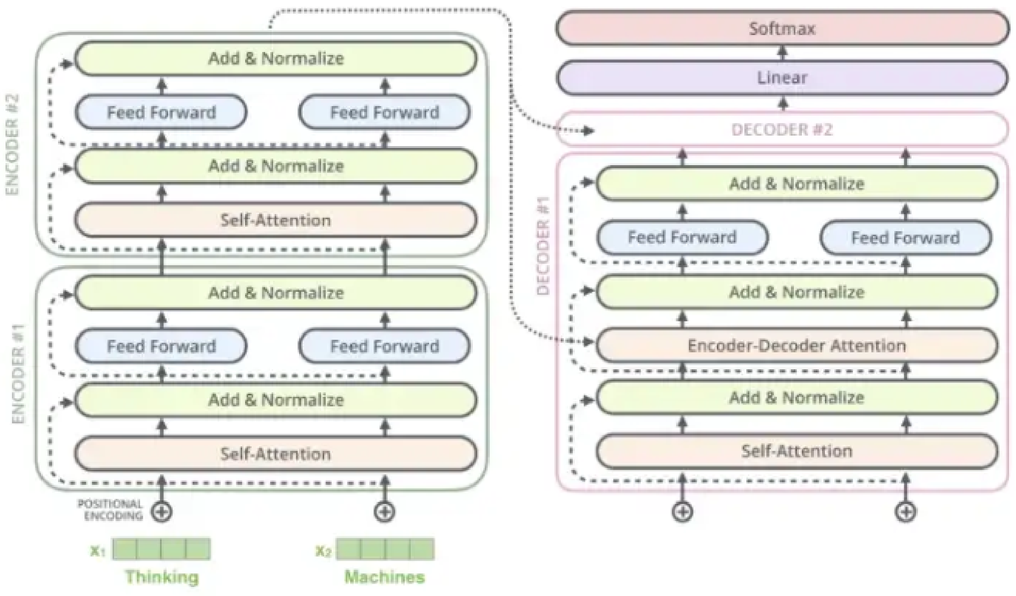

Figure 2: Transformer architecture of T5 *Source:Chen*

In terms of performance, T5 has been shown to be effective at a wide range of natural language processing tasks, and has achieved state-of-the-art results on several benchmarks. However, it is not necessarily better than BERT at all tasks, as the performance of a specific model depends on the specific task and dataset it is being applied to. In general, T5 is more flexible and can be used to perform a wider range of tasks than BERT, but it may require more computational resources and may not perform as well as task-specific models on certain tasks.

GPT-2 Radford et al. [2019] and GPT-3 are famous transformer based models, which exhibit competitive zero shot capabilities. GPT-2 was trained on 8 million web pages which led it to be trained on 1.5 billion parameters. GPT-2 takes the sequential input and can predict the next word in the sentence based on all the previous words. It is very modular and can be run on new datasets to generate conditional responses to part of sentences that the model hasn't seen previously because of the large data it has been trained on. GPT-3 which is much more powerful version and an iterative upgrade over GPT-2 is also a transormer model that has 175 billion parameters. It also calculates the next word that can appear in the sentence based on the highest probability of the words on which it has been trained on. GPT-3 is accessed by making API calls under the LmaS basis (language-model-as-a-service).

We intend to use the literature review done above, for our proposed methodology. We explore the both and attack and defense perspectives. By doing that we intend to come up with robust solution to cater both the sides.

3 Methodology

In our project we planned to implement a two step process and have eventually extended our newly proposed methodology to a three step approach. As the *first step*, for the poison data generation, we use the *Text-to-Text Transfer Transformer* (T5) to generate paraphrased sentences having the same structure from the original clean OLID and SST-2 data set. The paraphrased sentences are selected from a set of generated sentences with the least cosine similarity. This is because, as mentioned in our base paper, it is to be made sure that the generated poison data is far as possible from the clean input data. By this we make sure that the FTR is not increases. We then use the syntactic structure specified by Qi et al. [2021b], to change all inputs to the same sentence structure. After generating our poison data, we train our victim model F_θ, with the poisoned and clean dataset resulting in a backdoor model Kurita et al. [2020]. For this purpose we use the pre-trained T5 model available on Hugging Face Golla. This T5 model is trained to produce high quality , paraphrases.

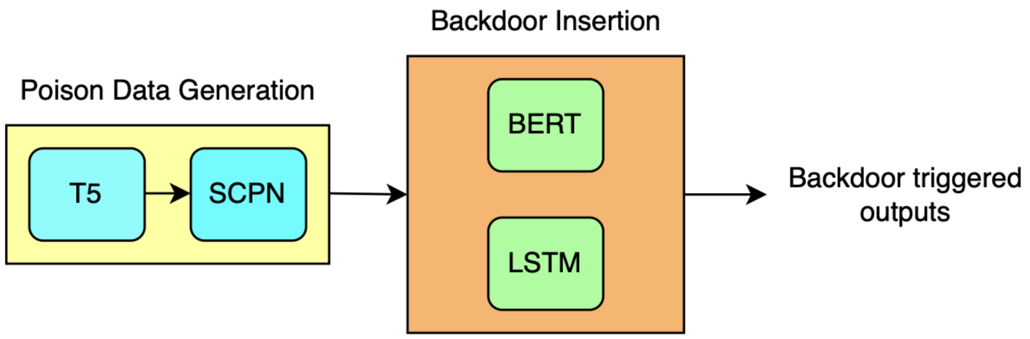

Figure 3: Syntactic Backdoor Attack

The *second step* of our approach , is to effectively reduce the False Negative cases i.e the *FTR*. This invloves the case where in benign input might contain portions of poisoned inputs which might trigger a backdoor response and might notify the user or developers, which reduces the stealthiness of the backdoored model making them unstable. In this step we try to provide stable activation/triggers for our backdoors. One of the effective way to make the backdoor's activation not affected by sub-sequences is to introduce negative data augmentation Yang et al. [2021b]. Besides creating poisoned samples inserted with the complete trigger sentence, we can further insert these sub-sequences into some clean samples without changing their labels to create negative samples. However, inducing some embeddings to the input might make our model less stealthier, which pushed us to other option, which uses the sytactic structure. We have integrated this step as a part of our poison data generation. We do this by taking the cosine similarity between input and the generated outputs and choose the output which has the least cosine similarity. This makes sure that additional stability, is given to the activation of the backdoors. This retrained model will be more stealthier than a model trained with just benign and poisoned inputs originally, making our attack more effective.

The above two steps explain our approach on effectively implement a stealthy attack. However as developers we need to see two sides of the coin , meaning come up with a better defense mechanism. Again we exploit the nature of our attack ,by retraining the model with paraphrased poisoned inputs. This methodology intuitively makes sense as by parphrasing the inputs and retraining it we make sure that the model exhibits a better defense nature.

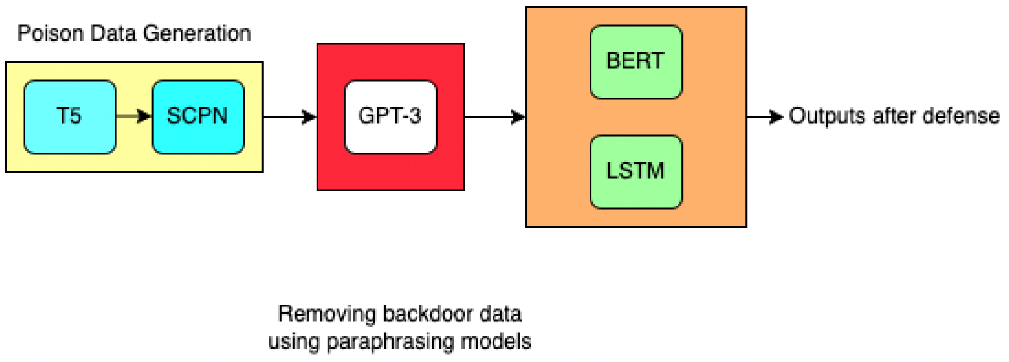

Figure 4: Effective Defense Mechanism implemented for Syntactical Attacks

There are multiple defense mechanisms proposed on Language Models for instance, Qi et al. [2020]. The *third step* of our approach is to prevent the triggering of backdoor and implement an effective defense methodology. We first assume that the model developer has the information about attacker's technique of inserting backdoor using syntactic templates. The developer can prevent this by retraining the model with the paraphrased poisoned data. In our case we pass the poisoned data through GPT-3's paraphrasing tool using OpenAI's API call and convert each sentences having a backdoor to another sentence having a similar meaning but a different structure. These inputs are used to retrain the model. This makes sure that the effect of backdoor inputs, for this particular attack is reduced and in turn enhancing the defensive nature of the model.

4 Model Setup

4.1 Datasets

We have experimented our above proposed methodology on the same data sets used in the base paper Qi et al. [2021b]. The text classification tasks including sentiment analysis, offensive language identification. The datasets are namely,

- Stanford Sentiment Treebank (SST-2) Socher et al. [2013].
- Offensive Language Identification Dataset (OLID) Zampieri et al. [2019].

We observed that the generating poison data for one of the initially proposed dataset AG's News Zhang et al. [2015] was taking more than 24 hours for just the poison data generation for training dataset alone. We plan to implement and explore this data, in our future work.

4.2 Victim Models

As of now we have executed the backdoor training and have evaluated the attacks on two models namely

- LSTM (BiLSTM)
- BERT Devlin et al. [2018].

4.3 Other Packages

For implementing our attacks in a smooth manner, we decided to use OpenAttack's SCPNAttacker and Textbugger. OpenAttack is a toolkit in python that can be used to adverserially attack different models which contains various flavors of textual based attacks THUNLP.

We have used the SCPNAttacker module to generate poisoned data, according to the syntactic templates. By using the OpenAttack tool we had the provision to, experiment with various other attacks. One of those attacks is Textbugger. As implied by it's name it introduces bugs into textual inputs like typo's or spelling mistakes that a user might make when they might be typing quickly. We experimented by adding the Textbugger to the poisoning pipeline after the SCPN attack which will add some typing error's meanwhile maintaining the syntactic format. We ran this new poisoned version of OLID and SST-2 on BERT model and found out some erratic behaviour in which the data would randomly trigger the backdoor and we were getting 50% accuracy, which might be due to the fact that the model has become robust and resistant to adversarial examples.

5 Results

In the first leg of our project, we worked on implementing the syntactical textual backdoor attack, explained in Qi et al. [2021b] . We used poisoned data available to attack the victim models BERT and LSTM. We trained the model for upto 10 epochs, by which it could be observed that the attack success rate reached to around 90%, for certain combination of dataset and victim models.

The Table-1 shows the Attack Success Rate *(ASR)* and Clean Accuracies *(CACC)*(%) derived from our implemented attack i.e., ***The First step and Second step*** in the proposed methodology. The results show that the *(ASR)* increased in case where the BERT is the victim model proving that our model performed better than the baseline results proposed in Qi et al. [2021b]. In case of LSTM as the victim model , it could be observed that our newly proposed methodology, exhibits more or less the same performance as the baseline results.

Dataset	Attack	LSTM		BERT	
	Method	ASR	CACC	ASR	CACC
OLID	SCPN	100	62.98	98.70	70.43
	T5 + SCPN	99.53	58.44	98.60	72.87
SST-2	SCPN	96.38	66.22	73.35	85.94
	T5 + SCPN	96.73	52.93	90.93	83.03

Table 1 : The above table shows the Attack Success Rate *(ASR)* and Clean Accuracies *(CACC)* (in %) of our implemented attack methodology **T5 + SCPN** on different datasets and victim models.

The Table-2 shows the Attack Success Rate *(ASR)* and Clean Accuracies *(CACC)* (%) derived for our defense mechanism for the proposed attack methodology i.e., ***The Third step*** in the proposed methodology. It could be observed that irrespective of the attack , the defense methodology we proposed worked efficiently in bringing the ASR down, meaning the defense is more effective on these type of attacks. This intuitively makes sense given the powerful performance of GPT3.

Dataset	Attack	LSTM		BERT	
	Method	ASR	CACC	ASR	CACC
OLID	SCPN	61.93	61.11	83.38	72.48
	T5 + SCPN	36.90	66.93	80.20	72.52
SST-2	SCPN	-	-	72.03	84.12
	T5 + SCPN	79.57	49.86	83.36	84.18

Table 2 : Results after using **GPT-3** defense of our implemented attack methodology **T5 + SCPN** on different datasets and victim models

One other common observation from all of our experiments is that the Clean Accuracies *(CACC)(%)* did not vary much, from the baseline clean accuracies, which included implementing our three step approach.This proves that the the essence of the backdoor attacks is preserved and our methodologies does not affect the performance of the models on clean data.

6 Conclusion

The inception point of our project came from the problem statement, of how we could improve an attack's performance by not compromising on the stealthiness aspect. We extended the methodology published in the Qi et al. [2021b], by optimising the poison data generation step. We do that by implementing a better pipeline , by integrating the poison data generation process with the pre-trained T5. By doing these we observed a better quality of poison data and the *(ASR)* increased or remained same for the given combinations of victim model and the dataset. This step also takes care of reducing the *(FTR)* by choosing the poisoned data, from the set of generated paraphrases, having the minimum cosine similarity, with the input. Further we tried to counter-attack our methodology by implementing a defense mechanism, in which we try to reduce the effect of backdoored triggers, eventually making our model immune to to the attack. By this we observed considerable decrease in the *(ASR)*, proving the effectiveness of our defense mechanism.

7 Future Work

As mentioned in the above sections we would like to implement our attack on a bigger dataset which AG,IMDB etc., and measure its effectiveness. Also, we tried to use OpenAttack's Texbugger to add more backdoor features into the data set which will reduce the FPR that could arise and trigger the backdoor. We plan to use portions from the Textbugger library like visual queues where we replace letters with similar looking numbers or other letters. Or we can also replace some letters with other keystrokes that lies in the surrounding of the key which might exist due to user's typing errors and could be overlooked by models and developers at a first glance.

8 Codebase Link

The project implementation can be found at `https://github.com/nyu-ce-projects/stealthy-syntactical-backdoor-attack`.

References

Xinyun Chen, Chang Liu, Bo Li, Kimberly Lu, and Dawn Song. Targeted backdoor attacks on deep learning systems using data poisoning. *arXiv preprint arXiv:1712.05526*, 2017.

Tianyu Gu, Brendan Dolan-Gavitt, and Siddharth Garg. Badnets: Identifying vulnerabilities in the machine learning model supply chain. *arXiv preprint arXiv:1708.06733*, 2017.

Wenkai Yang, Lei Li, Zhiyuan Zhang, Xuancheng Ren, Xu Sun, and Bin He. Be careful about poisoned word embeddings: Exploring the vulnerability of the embedding layers in nlp models. *arXiv preprint arXiv:2103.15543*, 2021a.

Yiming Li, Yong Jiang, Zhifeng Li, and Shu-Tao Xia. Backdoor learning: A survey. *IEEE Transactions on Neural Networks and Learning Systems*, 2022.

Fanchao Qi, Yangyi Chen, Xurui Zhang, Mukai Li, Zhiyuan Liu, and Maosong Sun. Mind the style of text! adversarial and backdoor attacks based on text style transfer. *arXiv preprint arXiv:2110.07139*, 2021a.

Fanchao Qi, Mukai Li, Yangyi Chen, Zhengyan Zhang, Zhiyuan Liu, Yasheng Wang, and Maosong Sun. Hidden killer: Invisible textual backdoor attacks with syntactic trigger. *arXiv preprint arXiv:2105.12400*, 2021b.

Wenkai Yang, Yankai Lin, Peng Li, Jie Zhou, and Xu 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 (Volume 1: Long Papers)*, 2021b.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

Mohit Iyyer, John Wieting, Kevin Gimpel, and Luke Zettlemoyer. Adversarial example generation with syntactically controlled paraphrase networks. *arXiv preprint arXiv:1804.06059*, 2018.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, Peter J Liu, et al. Exploring the limits of transfer learning with a unified text-to-text transformer. *J. Mach. Learn. Res.*, 21(140):1–67, 2020.

Ramsri Goutham Golla. t5-large-paraphraser-diverse-high-quality. `https://huggingface.co/ramsrigouthamg/t5-large-paraphraser-diverse-high-quality`.

Qiurui Chen. T5ArchitectureReference. `https://medium.com/analytics-vidhya/t5-a-detailed-explanation-a0ac9bc53e51`.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9, 2019.

Keita Kurita, Paul Michel, and Graham Neubig. Weight poisoning attacks on pre-trained models. *arXiv preprint arXiv:2004.06660*, 2020.

Fanchao Qi, Yangyi Chen, Mukai Li, Yuan Yao, Zhiyuan Liu, and Maosong Sun. Onion: A simple and effective defense against textual backdoor attacks. *arXiv preprint arXiv:2011.10369*, 2020.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D Manning, Andrew Y Ng, and Christopher Potts. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 conference on empirical methods in natural language processing*, pages 1631–1642, 2013.

Marcos Zampieri, Shervin Malmasi, Preslav Nakov, Sara Rosenthal, Noura Farra, and Ritesh Kumar. Predicting the type and target of offensive posts in social media. *arXiv preprint arXiv:1902.09666*, 2019.

Xiang Zhang, Junbo Zhao, and Yann LeCun. Character-level convolutional networks for text classification. *Advances in neural information processing systems*, 28, 2015.

THUNLP. OpenAttack. `https://github.com/thunlp/OpenAttack`.