Argot: Generating Adversarial Readable Chinese Texts

Zihan Zhang[1*] , **Mingxuan Liu**[1*] , **Chao Zhang**[1,2†] , **Yiming Zhang**[1] ,
Zhou Li[3] , **Qi Li**[1,2] , **Haixin Duan**[1,2,4] , **Donghong Sun**[1,2]
[1]Tsinghua University
[2]Beijing National Research Center for Information Science and Technology
[3]University of California Irvine
[4]Qi An Xin Group
{zhangzih19,liumx18}@mails.tsinghua.edu.cn, chaoz@tsinghua.edu.cn,
zhangyim17@mails.tsinghua.edu.cn, zhou.li@uci.edu {qli01,duanhx,sundh105}@tsinghua.edu.cn

Abstract

Natural language processing (NLP) models are known vulnerable to adversarial examples, similar to image processing models. Studying adversarial texts is an essential step to improve the robustness of NLP models. However, existing studies mainly focus on analyzing English texts and generating adversarial examples for English texts. There is no work studying the possibility and effect of the transformation to another language, e.g, Chinese. In this paper, we analyze the differences between Chinese and English, and explore the methodology to transform the existing English adversarial generation method to Chinese. We propose a novel black-box adversarial Chinese texts generation solution Argot, by utilizing the method for adversarial English samples and several novel methods developed on Chinese characteristics. Argot could effectively and efficiently generate adversarial Chinese texts with good readability. Furthermore, Argot could also automatically generate *targeted* Chinese adversarial text, achieving a high success rate and ensuring readability of the Chinese.

1 Introduction

Deep Neural Networks (DNNs) that process images are known to be vulnerable to adversarial examples. Recent works showed that Natural Language Processing (NLP) with DNNs is also vulnerable [Li *et al.*, 2018a]. Given the fact that NLP plays an important role in information processing, e.g., sentiment analysis of reviews used in recommendation systems [Ravi and Ravi, 2015] and toxic content identification in online governance [Nobata *et al.*, 2016], studying adversarial texts of NLP models is crucial and an essential step to improve NLP models' robustness.

However, generating adversarial texts is more challenging than generating adversarial images. Image data is continuous, but text data is discrete. Even minor perturbations applied to a word vector could yield a non-existed word vector (i.e., not associated with any valid word in the language). In addition, These non-existed word vector will result in meaningless texts, decreasing the perturbations may affect the readability of texts. As a result, adversarial example generation solutions for image processing DNNs cannot be directly ported to NLP DNNs.

Some works have been proposed to generate adversarial texts [Li *et al.*, 2018a]. But they all target English texts. Researchers [Li *et al.*, 2018a; Ebrahimi *et al.*, 2017] proposed five commonly used methods to generate English adversarial texts in black-box scenario, i.e., insert, delete, swap, substitute-C and substitute-W. But none of them discussed the possibility about transferring the proposed methods to Chinese. Moreover, Chinese itself has some unique characteristics. Therefore, two research questions are raised: (1) Can existing adversarial English text generation methods be transformed to Chinese? How effective are they? (2) Are there new methods for generating adversarial texts based on the characteristics of Chinese?

In this paper, we analyzed the difference between English and Chinese, and found Chinese texts have three unique linguistic characteristics: pronunciation (*pinyin*), vision perception (*glyph*) and composition of characters (*radicals*). We therefore proposed a novel adversarial Chinese text generation solution Argot, by adding perturbations based on these characteristics. This solution could also work on other languages which also have similar characteristics as Chinese, e.g., Korean and Japanese. In addition, Argot is able to generate both *targeted* and *non-targeted* Chinese adversarial text.

We have evaluated Argot's performance of success rate, perturbation, time consumption and readability, on several NLP models for both targeted and non-targeted attacks. The results showed that, in average, Argot could generate adversarial texts with a success rate over 97.7% for non-targeted attack, by only introducing less than 11.6% perturbations to original texts, and 98.73% success rate with 12.25% perturbations for targeted attack. The low perturbation rate implies that, the adversarial texts generated by Argot have a good readability, which is also confirmed by a user study conducted by us. Furthermore, we proposed several candidate defenses against the attacks, and evaluated Argot's performance against these defenses.

[*] These two authors contributed equally

[†] Corresponding author

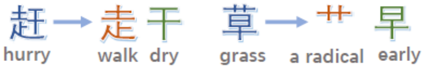

Figure 1: Illustration of splitting characters into radicals.

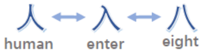

Figure 2: Illustration of characters with similar glyph.

2 Background

2.1 Chinese Text

Similar to other languages like English, sentence is the basic unit to express a complete idea in Chinese. In a sentence, the smallest *sememe* is character in Chinese, while word is the *sememe* in English. In Chinese, a few characters make up a word (phrase). English words in a sentence are separated by the separator `space`. But no separators exist between Chinese characters and words, and thus an extra segmentation step is often required for Chinese NLP tasks.

Chinese word. A Chinese word is typically made up of two to four characters, while an English word usually consists of a string of alphabet letters. While changing, adding or deleting a letter usually does not impact the readability of an English word, doing so on characters can completely change the meaning and readability of a Chinese word. For example, if we delete the character 不(not) from the word 不好(bad), the remaining word 好(good) expresses an opposite meaning.

Chinese character. A Chinese character is typically composed of two radicals, as shown in Figure 1, while the English sememe (word) is composed of alphabet letters. Inserting/deleting/changing a letter in a English word will yield a new word which could be printed out and easily understood, e.g., f oolish/folish/fo0lish for "foolish" [Li *et al.*, 2018a]. However, changing/adding/deleting a radical inside a character will yield either a character with very different appearances, or a character which does not exist in Chinese character dictionary and cannot print or input to DNNs. As a result, creating a new *sememe* in Chinese (i.e., character) acceptable by readers is much harder than English (i.e., word).

Creating similar Chinese word. We figured out two unique Chinese linguistic features can be utilized to generate Chinese words without affecting the readability much.

Similar glyph. Chinese is a type of hieroglyphics language, which utilizes the image perception of pictograph (glyph) to help people understand a character or sentence [Wu *et al.*, 2019]. Some characters are similar in glyph, as shown in Figure 2. It is quite likely that a native Chinese speaker can comprehend a sentence in the same way even when a character is transformed to another one with similar glyph. Moreover, as most radicals are also legitimate characters, separating a character into multiple radicals usually will not degrade the readability much, as shown in Figure 1.

Similar sound. Chinese is also a phonetic language with *pinyin* (a romanized spelling) and tone to distinguish the pronunciation between words. As shown in Figure 3, the English letters on top of each Chinese character are its pinyin representations. Each character also has one out of four tones. Many characters or words share similar pinyin and tones. Replacing a word in a sentence with another one of similar pronunciation will not change its readability to a Chinese native speaker. The major reason is that, Chinese users usually use pinyin input method to type Chinese characters, but oftentimes select a wrong character of same pinyin or of different pinyin (e.g., due to regional dialect), making Chinese readers familiar with the typos.

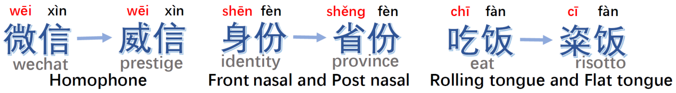

Figure 3: Illustration of similar pronunciations in Chinese words.

2.2 Adversarial Perturbations to Text

The adversary's goal is generating adversarial texts by perturbing a piece of Chinese text to mislead a target NLP application (e.g., toxic content detection), while keeping the texts comprehensible by readers and the meaning intact. While English is still the most popular language served by online NLP services, e.g., Google Cloud NLP and Microsoft Azure Text Analytics, services supporting Chinese are catching up. Whether they can defend against adversarial perturbations is questionable, as their backend technique is based on DNN, which is known to be vulnerable to such attacks.

Here we formalize the problem. Assume an NLP classifier f classifies a text piece $x \in \mathcal{X}$ into a label $y \in \mathcal{Y}$, i.e., $y = f(x)$. The adversarial perturbation function h changes x to $x_a \in \mathcal{X}$ i.e., $x_a = h(x)$. The adversary aims to change its prediction label such that $f(x_a) = z, z \neq y$. The change can be *targeted* ($z = s$, where $s \in \mathcal{Y}$ and s is pre-defined by the adversary) or *non-targeted* ($z \in \{\mathcal{Y} - y\}$). Also, we need x_a preserve the meaning of x from the perspective of readers.

Threat Model. In this work, we assume that the adversary launches *blackbox* attack, who does not know the architecture, training dataset, parameters and hyper-parameters of classifier f. All they can do is querying the public API of f using $x, x_a \in \mathcal{X}$ to obtain the output label and its confidence score. While previous works focus on whitebox attack [Gong *et al.*, 2018], we believe *blackbox* attack is more practical as model owner usually customizes a public model (e.g., retrain or tuning hyper-parameters). While the adversary's knowledge is much more limited in this setting, we found our attack can still achieve high accuracy.

3 Generating Adversarial Chinese Text

3.1 Adversarial Changes in Chinese Word

Previous works about generating adversarial text focused on English [Li *et al.*, 2018a] and five types of word perturbation have been proposed: 1) **Insert** a space into the word; 2) **Delete** a random letter; 3) **Swap** random two adjacent letters; 4) **Substitute-C (Sub-C)** or replace characters with visually similar characters; 5) **Substitute-W (Sub-W)** or replace the word with its top_k nearest neighbors in a context-aware word vector space. We examine those methods but found not all of them can be transformed to Chinese. For example, deleting a character or radical in Chinese will have much more impact to comprehension comparing to English. In the end, we found five types of perturbations applicable to Chinese while 2 of them are transformed from attacks against English with some adjustment and 3 are novel and unique to Chinese[1]. Due to the fact that languages in East Asia like Korean and Japanese share similar properties, e.g., hieroglyphics, our methods are expected to be applicable to those languages as well.

Synonyms. This method is transformed from English **Sub-W** method. But we focus on synonyms defined in a vocabulary without using word embedding.

Shuffle. This method is transformed from English **Swap** method. We apply it to characters within a Chinese word.

Splitting-character*. As aforementioned, some radicals are also characters. Therefore, a special method for generating adversarial Chinese texts is splitting characters into radicals. In particular, we only split characters of left-right or semi-enclosed radical structures (left and right part of Figure 1, respectively), as it causes less confusion to readers. This method is similar to **Insert** spaces to English words.

Glyph*. As aforementioned, characters with similar glyph can be understood smoothly in a sentence. Therefore we can replace characters with ones of similar glyph. Note that, this is different from **Sub-C**, as the candidate glyph pairs are much more than English letters (e.g., only 3 options, `1-l`, `O-o` and `a-@`, are explored in [Li *et al.*, 2018a]).

Pinyin*. As aforementioned, replacing a character with another one of similar *pinyin* also yield readable texts. This feature is unique to hieroglyphics languages, and could be utilized to generate adversarial texts as well.

3.2 Argot

We propose a solution named Argot to generate adversarial Chinese texts, by applying aforementioned perturbations to target Chinese words in a sentence. The workflow of Argot is shown in Figure 4.

Given a Chinese sentence (termed x) and a target NLP model M, Argot first segments x into a list of Chinese words ($W =< w_1, w_2, ..., w_n >$) using a standard NLP tool[2]. To retain the readability, we only mutate a limited set of important words $W_{imp} \subset W$. Specifically, we sort each word's contribution to the classification result of text x under the NLP classifier f of target NLP model M, and mark important words accordingly. Then, we iterate important words W_{imp} in order and apply aforementioned perturbations to each word. The target model M will be queried with each new sentence. If M yields an expected label, the query sentence is reported as an adversarial example. Details are presented as follows.

Querying model M. When evaluating the importance of a word or the label of a mutated sentence, we will query the target NLP model M. Given an input x, we assume M returns the confidence score for *every* label, and outputs a label y if and only if the confidence score $Score_f(x, y)$ is highest among all candidate labels. This is the normal setting for MLaaS APIs and a common assumption of previous blackbox attacks [Li *et al.*, 2018a]. When evaluating the importance of a word, we remove the word from the sentence, and query the target model M with the new sentence to compute the decrease of confidence. When evaluating the label of a mutated sentence, we design two score functions L_u and L_t to guide targeted and non-targeted attacks respectively.

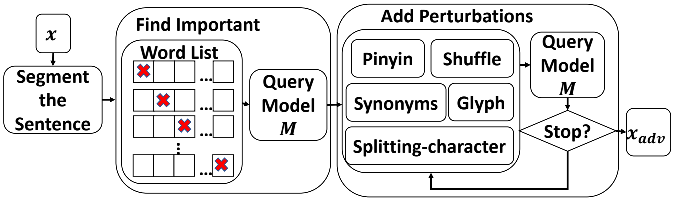

Figure 4: Overview of Argot's workflow.

Non-targeted attack. After adding perturbations on the sentence x, a new sentence x_a is yielded, where $x_a = h(x)$. We monitor the drops of confidence score $L_u = Score_f(x, y) - Score_f(x_a, y)$ to evaluate perturbations' effects. Argot will iteratively mutate the target sentence, to generate a sentence x_a^n after n iterations, whose confidence score $Score_f(x_a^n, y)$ drops to a very low level and is even lower than some other lable's (e.g., z) confidence score $Score_f(x_a^n, z)$.

Targeted attack. In this setting, we use the increase rather than decrease of target label s' confidence score $L_t = Score_f(x_a, s) - Score_f(x, s)$, to measure the effectiveness of perturbations. The ultimate goal is that, after multiple iterations, the label s' confidence score $Score_f(x_a^n, s)$ reaches to a very high level and is higher than all other labels'.

Finding important words. As Argot is a blackbox attack, gradients are inaccessible so we use the output label and confidence score as alternative indicators. In particular, for each word w_i in W, we delete it from the original sentence, query the new sentence with M, and compare the result to the original one's. If deleting w_i makes the original label's confidence score drop more than deleting another word, then w_i is more important than the other. After enumerating W, we get a list of words sorted by importance.

Adding perturbations. Given a seed sentence, one round of mutations will be performed on this sentence as follows. The important words will be iterated one by one in the decreasing order of importance. In each iteration, the selected word will be mutated with five candidate perturbations, yielding five new sentences which will be queried with M. If either new sentence satisfies the ultimate goal of non-targeted or targeted attack, Argot will stop. Otherwise, this round stops after all important words have been iterated. And, the yielded sentence with highest L_u or L_t is chosen as the new seed sentence, and a new round of mutations will start. Among the five perturbations, splitting-character, synonyms and shuffle are straightforward, so we only elaborate the remaining two as follows.

Pinyin. Homophone and typos in typing pinyin can be utilized to generate adversarial words with little influence on readability. We propose to replace the target word with the following three types of words:

[1] We use "*" to mark those novel methods.

[2] https://github.com/fxsjy/jieba

Algorithm 1 : The detail of function glyph(c).

```
Input: Original character $c$.
Output: New character $c_2$ with similar glyph.
Read in all Chinese radicals into a list $all\_radicals$.
$similarity = 0, c_2 = c, candidates = \{\}$
$radicals \leftarrow$ decompose_radicals$(c)$
for $radical \in radicals$ do                          ▷ Replace radicals
    for $other \in all\_radicals$ do
        if $other \neq radical$ then
            $radicals' \leftarrow$replaceWith$(radical, other)$
            $candidates \leftarrow candidates \cup$ Val $(radicals')$
        end if
    end for
end for
for $radical \in radicals$ do                          ▷ Delete radicals
    $radicals' \leftarrow$deleteFromRadicals$(radical)$
    $candidates \leftarrow candidates \cup$ Val $(radicals')$
end for
for $other \in all\_radicals$ do                       ▷ Add radicals
    $radicals' \leftarrow$addIntoRadicals$(other)$
    $candidates \leftarrow candidates \cup$ Val $(radicals')$
end for
for $c_1 \in candidates$ do              ▷ Assess the similarity of candidates
    $score \leftarrow$siamese_similarity$(c, c_1)$
    if $score > similarity$ then
        $similarity = score, c_2 = c_1$
    end if
end for
return $c_2$;
```

- A homophone word [Hiruncharoenvate *et al.*, 2015], which has a same pinyin representation.
- A word that has a similar pinyin representation but different head and tail nasal, by interchanging `an` and `ang`, `in` and `ing`, as well as `en` and `eng` in the pinyin representation.
- A word that has a similar pinyin but different rolling/flat tongues, by interchanging `c` with `ch`, `z` with `zh`, and `s` with `sh` [Chen and Lee, 2000].

Glyph. For a given word consisting of multiple characters, we replace some characters with ones of similar glyph. The core challenge is finding similar characters. For each character, we first decompose it into a set of radicals, with an open source tool[3]. Then, we update the radical set with three strategies: **replacing** a radical with another one from the Chinese vocabulary, **deleting** one radical, or **adding** another radical. A special function **Val()** is utilized to check whether the new radical set can make up a legitimate Chinese character. If yes, it returns the yielded character, otherwise it returns *NULL*. In this way, we could generate a set of characters with similar glyph. The character that is most similar to the original character will be used to mutate the sentence. Algorithm 1 shows the details. To measure the similarity between two characters, we develop a DNN model based on *siamese* network, which has been found effective in comparing input data [Chopra *et al.*, 2005]. The structure is shown in Figure 5. It takes two pictures I_1, I_2 of characters as input and sends them to two identical CNN models g with parameters θ. In our prototype, the model g consists of three convolutional layers and each is followed by a max-pooling layer. The number of convolutional filters is 64 for the first layer and 128 for last two. Then, the distance between two pictures' features $g(I_1; \theta)$ and $g(I_2; \theta)$ is computed, and a confidence score is produced via *sigmoid*. We selected 16,008 pairs of similar characters from an open source corpus[4] as the training data. After training, it can calculate the similarity between any two characters.

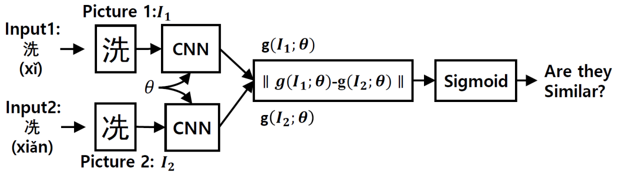

Figure 5: Architecture of Siamese network.

4 Evaluation

4.1 Evaluation Setup

NLP task. We select one of the most common NLP tasks, i.e., text classification, as the NLP application to be attacked. We use sentiment classification (2-class) to evaluate the non-targeted attack, and use news classification (5-class) to evaluate the targeted attack.

Dataset. We create a dataset for sentiment classification using samples from an existing Chinese NLP dataset[5]. We choose the reviews of hotel, book, electronic product that are crawled from e-commerce websites. Our dataset is composed of 15,471 negative reviews and 16,188 positive reviews. We manually check all these reviews and delete the ones that are too short or have ambiguous labels. We divide this dataset into 80% and 20% for training and validation respectively. For news classification, we use the THUNews dataset [Sun *et al.*, 2016]. Out of the fourteen classes, we select five classes from the dataset, i.e., affair, education, finance, society and sports. For each class, we sample 25,000 texts as training set and 5,000 as validation set.

Target Models. We choose two existing models built on top of *CNN* [Kim, 2014] and *LSTM* [Zhang *et al.*, 2015] as the target models. They are widely used for many NLP tasks. As aforementioned, a big difference between English and Chinese is that, there is a word separator (space) in English, but not in Chinese. As a result, Chinese NLP models usually have two variants, i.e., character-based and word-based, depending on whether the model performs word segmentation and whether it performs characters or words embedding. We consider both variants in evaluation.

Metrics. Three metrics are used to comprehensively evaluate the generated adversarial texts: success rate, perturbations and efficiency. *Success rate* reflects how many sentences in the validation set have been successfully mutated to

[3] https://github.com/kfcd/chaizi/blob/master/chaizi-jt.txt

[4] https://github.com/zzboy/chinese

[5] https://github.com/SophonPlus/ChineseNlpCorpus

	Target	Affair			Education			Finance			Society			Sports		
Origin		S	P	T	S	P	T	S	P	T	S	P	T	S	P	T
Affair	char				100%	0.16	0.37	99%	0.22	0.51	100%	0.10	0.29	99%	0.23	0.55
	word				100%	0.08	0.20	100%	0.02	0.14	100%	0.01	0.11	96.8%	0.14	0.27
Education	char	98.4%	0.25	0.52				90%	0.32	0.69	99.2%	0.18	0.39	84%	0.37	0.79
	word	99%	0.04	0.13				100%	0.02	0.12	100%	0.02	0.11	96.6%	0.10	0.23
Finance	char	100%	0.12	0.35	99.4%	0.15	0.34				99.6%	0.12	0.30	98.4%	0.24	0.55
	word	99.8%	0.04	0.14	100%	0.05	0.17				99.8%	0.02	0.11	97%	0.10	0.24
Society	char	100%	0.15	0.42	99.8%	0.15	0.38	99.8%	0.22	0.53				99.4%	0.21	0.50
	word	100%	0.04	0.17	100%	0.06	0.21	100%	0.03	0.17				100%	0.08	0.24
Sports	char	99%	0.17	0.41	99.2%	0.16	0.39	96.6%	0.28	0.59	99.6%	0.15	0.36			
	word	100%	0.02	0.14	100%	0.04	0.17	100%	0.03	0.16	99.8%	0.01	0.10			

Table 1: Targeted attack results of Argot (S: success rate, P: perturbation, T: Time/Char.).

Model	Accuracy	S	P	T
char-based CNN	89%	93.73%	0.14	0.28
word-based CNN	90%	98.31%	0.09	0.21
char-based LSTM	90%	99.73%	0.13	0.39
word-based LSTM	90%	99.05%	0.09	0.58

Table 2: Non-targeted attack results of Argot (S: success rate, P: perturbation, T: Time/Char.).

fool the target NLP model. *Perturbation* reflects the average percentage of characters in a successful adversarial sentence that have been mutated. It also reflects the impact to readability, because more perturbation tend to make readability worse. *Efficiency* represents the average time consumed by Argot when generating adversarial texts. Longer input texts usually costs more time, since Argot is likely to mutate more words. As a result, we use the metric *Time/Char.*, i.e., the time cost divided by the text length (seconds per character).

Implementation Details. The word segmentation tool for Chinese text pre-processing used in our experiment is jieba[6]. The word embedding scheme is trained from the Chinese wiki corpus [Li *et al.*, 2018b] with an embedding dimension of 300, using *word2vec* [Mikolov *et al.*, 2013]. To generate perturbations, we use a tool [Tiantian, 2015] to transform Pinyin to character and another tool [Hai Liang Wang, 2018] to transform character to Pinyin. In addition, we use a tool [Hai Liang Wang, 2017] to choose the appropriate synonym for a word.

4.2 Evaluation Results

Non-targeted Attack. Table 2 shows the results of applying all 5 perturbations for non-targeted attack. The second column shows the accuracy of original black-box models (all around 90%). The best success rate is 99.73% on the char-based *LSTM*, while the worst case can still reach 93.73% on char-based *CNN*. The forth column shows the percentage of perturbations added by Argot comparing to the text length, showing that minor mutations are sufficient to yield the desired adversarial texts.

Targeted Attack. Table 1 shows the results of target attack. In most cases, Argot achieves more than 95% success rate by introducing only a few perturbations. The small perturbation rate also implies that Argot is able to retain semantics of the original sentences.

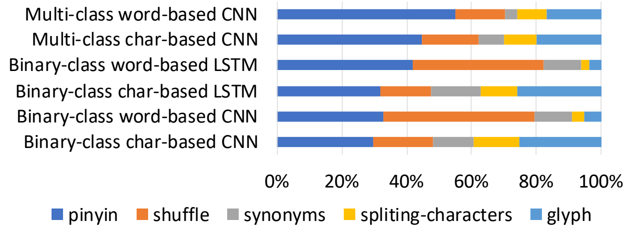

Figure 6: The distribution of 5 perturbations.

Efficiency Analysis. From Table 2 and Table 1, we can see that Argot is fast at generating adversarial texts. In the best case, it can process one Chinese character within 0.1 second on average in the target attack setting, and 0.21 second for non-target attack. Note that, the legth of input texts in the target attack dataset are longer than non-targeted attack dataset in average (1000 for targeted and 80 for non-targeted), which increases the time of generating adversarial texts. Also, the efficiency of Argot is strongly related to the specific attack target. For example, adversarial perturbation from *Affair* to *Society* is much more efficient comparing to that from *Education* to *Sports*. We speculate the root cause is that the categories *Affair* and *Society* are very similar and overlapped to some extent, which makes targeted attack easier, but the categories *Education* and *Sports* are very different.

Contributions of Perturbations. For all adversarial texts, we evaluated what perturbations are added to the original texts. The distributions of these 5 perturbation methods are shown in Figure 6. As we can see, the method *pinyin*, *glyph* and *shuffle* account for the largest proportion. In particular, *pinyin* accounts for the most perturbations overall, which can be explained by the large perturbation space under this method. *Pinyin* and tone determine the pronunciation of a character directly, and they also contains semantic meanings that cannot be provided by the writing system. Existing Chinese NLP models focus on achieving robust classification results on words and characters, thus they are vulnerable to *pinyin* perturbations. Other than *pinyin*, glyph contributes more among the char-based models and shuffle contributes more for word-based models, mainly because those models concentrate on different set of features.

Readability Evaluation. The ultimate goal of adversarial texts is to mislead NLP models but not humans. So, we further evaluated the human readability of the generated adversarial texts, by conducting a user study. We randomly selected 50 adversarial texts generated by Argot, and queried

[6] https://github.com/fxsjy/jieba

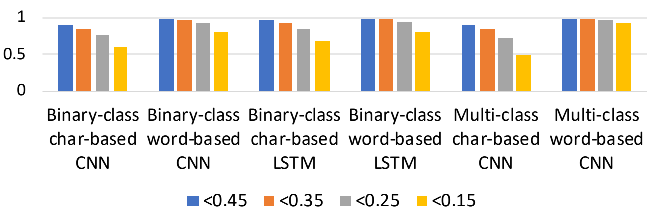

Figure 7: Relation of perturbations and success rate. The x-axis refers to perturbation threshold and models. The y-axis refers to the success rate of non-targeted and targeted attacks.

25 volunteer native Chinese speakers around the age of 24 with a gender ratio of 1:1. In the user study, for each adversarial text, volunteers are given the pair of original text and adversarial text, and three questions to answer: (1) whether the adversarial text is comprehensible, (2) what is the label of the adversarial text, and (3) the semantic quality score (0-5) of adversarial texts comparing to the original text. From the result of this survey, 84% volunteers think that the generated adversarial texts are readable. 91% of them consider the label of the generated adversarial texts same as the original texts, implying that the adversarial texts do not change the semantics significantly. The average score of semantic quality is 4.6, very close to the highest score 5. In summary, we believe that the generated adversarial texts are of very high quality in terms of readability and semantic understanding.

Factors Affecting Success Rates. The first factor we look into is the perturbation rate. Figure 7 shows the changes of success rate given different perturbation rate threshold. We can see that, the success rate is decreasing when fewer perturbations are allowed in all attack settings. Furthermore, the success rate keeps at a relatively high level even we reduce the perturbation threshold to a relatively low level. Intuitively, the more perturbations added to the original text, the greater the impact on readability will be. As a result, we can trade perturbation rate with success rate to get adversarial texts of higher quality. In addition, from Table 1 and Table 2, we can see that the success rates against word-based models are higher than that of char-based models. It suggests that char-based Chinese NLP models in general are more robust than word-based NLP models against adversarial perturbations. Another factor that may affect the success rate is the length of the original text. Given a longer input text, Argot could generate adversarial texts with a higher success rate, probably because the longer input texts leave more opportunities for Argot to add perturbations.

Defenses. Based on the perturbation methods Argot used, we proposed two defense solutions.

- *Embedding Pinyin Features.* The robustness of Chinese NLP models could benefit from adding information from pinyin [Zhu *et al.*, 2018]. We therefore propose to integrate the *pinyin* embedding with word embedding to improve the robustness of the NLP models. Results showed that, this defense lowers the success rate of attacks from 70.1% to 65.4% on average.
- *Embedding Glyph Features.* Adding the glyph information to Chinese NLP models could also improve their robustness [Wu *et al.*, 2019]. Results showed that, this defense lowers the success rate from 80.1% to 77.7% on average, if Argot only uses glyph perturbations.

The results show that integrating pinyin and glyph features of Chinese texts indeed improves Chinese NLP models' robustness. However, the success rates of our attacks are still high. A better defense is needed and we leave it as our future work.

5 Related Work

Adversarial texts have been studied a lot in the literature. Most of them target English texts though. These solutions can be categorized into white-box and black-box settings.

White-box setting. Attackers mainly use gradient computed by the model to assess each word's impacts and add perturbations to important words to generate adversarial English texts [Papernot *et al.*, 2016; Li *et al.*, 2018a; Liang *et al.*, 2017; Samanta and Mehta, 2017; Gong *et al.*, 2018; Miyato *et al.*, 2017]. In addition, the interpretability of adversarial texts is studied recently [Sato *et al.*, 2018].

Black-box setting. This setting assumes the adversary can only access model's input and output. Gao et al. proposed TS (temporal score) and TTS (temporal tail score) [Gao *et al.*, 2018] to evaluate the importance of each word and add letter-level disturbance to the text. Kuleshov et al. [Kuleshov *et al.*, 2018] and Alzantot et al. [Alzantot *et al.*, 2018] use similar word in the vector space to find adversarial texts. Li et al. [Li *et al.*, 2018a] and Liang et al. [Liang *et al.*, 2017] proposed iterative approaches to mutate sentence from words and evaluated on real-world NLP applications. As Chinese text is composed in a different way (characters and radicals), we proposed a novel solution in this paper.

6 Conclusion

In this study, we analyze the unique characteristic of Chinese compared to English and the limitation of existing adversarial text generation solutions for English. Based on its characteristic, we propose a black-box adversarial Chinese text generation solution Argot for two attack scenarios, i.e., targeted attack and non-targeted attack. In Argot, we use five methods pinyin, glyph, splitting-characters, shuffle, and synonyms to add perturbations. We evaluate the three metrics (success rate, perturbation and efficiency) of generated texts of Argot. The results show that Argot could generate adversarial texts with high success rate and relatively small perturbations. Meanwhile, the generated texts can maintain good readability. According to user study, most volunteers can understand the meaning and get correct label of adversarial texts.

Acknowledgements

This work was supported in part by National Key Research and Development Program of China under Grant 2018YFB2101501 and 2016QY12Z2103, National Natural Science Foundation of China under Grant 61772308, 61972224, U1736209, U1836213 and U1636204 and BNRist Network and Software Security Research Program under Grant BNR2019TD01004 and BNR2019RC01009.

References

[Alzantot *et al.*, 2018] Moustafa Alzantot, Yash Sharma, Ahmed Elgohary, Bo-Jhang Ho, Mani Srivastava, and Kai-Wei Chang. Generating natural language adversarial examples. *arXiv preprint arXiv:1804.07998*, 2018.

[Chen and Lee, 2000] Zheng Chen and Kai-Fu Lee. A new statistical approach to chinese pinyin input. In *Proceedings of the 38th Annual Meeting of the Association for Computational Linguistics*, pages 241–247, 2000.

[Chopra *et al.*, 2005] Sumit Chopra, Raia Hadsell, Yann LeCun, et al. Learning a similarity metric discriminatively, with application to face verification. In *CVPR (1)*, pages 539–546, 2005.

[Ebrahimi *et al.*, 2017] Javid Ebrahimi, Anyi Rao, Daniel Lowd, and Dejing Dou. Hotflip: White-box adversarial examples for text classification. *arXiv preprint arXiv:1712.06751*, 2017.

[Gao *et al.*, 2018] Ji Gao, Jack Lanchantin, Mary Lou Soffa, and Yanjun Qi. Black-box generation of adversarial text sequences to evade deep learning classifiers. In *2018 IEEE Security and Privacy Workshops (SPW)*, pages 50–56. IEEE, 2018.

[Gong *et al.*, 2018] Zhitao Gong, Wenlu Wang, Bo Li, Dawn Song, and Wei-Shinn Ku. Adversarial texts with gradient methods. *arXiv preprint arXiv:1801.07175*, 2018.

[Hai Liang Wang, 2017] Hu Ying Xi Hai Liang Wang. Synonyms, 2017.

[Hai Liang Wang, 2018] Hu Ying Xi Hai Liang Wang. python-pinyin, 2018.

[Hiruncharoenvate *et al.*, 2015] Chaya Hiruncharoenvate, Zhiyuan Lin, and Eric Gilbert. Algorithmically bypassing censorship on sina weibo with nondeterministic homophone substitutions. In *Ninth International AAAI Conference on Web and Social Media*, 2015.

[Kim, 2014] Yoon Kim. Convolutional neural networks for sentence classification. *arXiv preprint arXiv:1408.5882*, 2014.

[Kuleshov *et al.*, 2018] Volodymyr Kuleshov, Shantanu Thakoor, Tingfung Lau, and Stefano Ermon. Adversarial examples for natural language classification problems. 2018.

[Li *et al.*, 2018a] Jinfeng Li, Shouling Ji, Tianyu Du, Bo Li, and Ting Wang. Textbugger: Generating adversarial text against real-world applications. *arXiv preprint arXiv:1812.05271*, 2018.

[Li *et al.*, 2018b] Shen Li, Zhe Zhao, Renfen Hu, Wensi Li, Tao Liu, and Xiaoyong Du. Analogical reasoning on chinese morphological and semantic relations. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 138–143. Association for Computational Linguistics, 2018.

[Liang *et al.*, 2017] Bin Liang, Hongcheng Li, Miaoqiang Su, Pan Bian, Xirong Li, and Wenchang Shi. Deep text classification can be fooled. *arXiv preprint arXiv:1704.08006*, 2017.

[Mikolov *et al.*, 2013] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*, 2013.

[Miyato *et al.*, 2017] Takeru Miyato, Andrew M. Dai, and Ian J. Goodfellow. Adversarial training methods for semi-supervised text classification. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*, 2017.

[Nobata *et al.*, 2016] Chikashi Nobata, Joel Tetreault, Achint Thomas, Yashar Mehdad, and Yi Chang. Abusive language detection in online user content. In *Proceedings of the 25th international conference on world wide web*, pages 145–153. International World Wide Web Conferences Steering Committee, 2016.

[Papernot *et al.*, 2016] Nicolas Papernot, Patrick McDaniel, Ananthram Swami, and Richard Harang. Crafting adversarial input sequences for recurrent neural networks. In *MILCOM 2016-2016 IEEE Military Communications Conference*, pages 49–54. IEEE, 2016.

[Ravi and Ravi, 2015] Kumar Ravi and Vadlamani Ravi. A survey on opinion mining and sentiment analysis: tasks, approaches and applications. *Knowledge-Based Systems*, 89:14–46, 2015.

[Samanta and Mehta, 2017] Suranjana Samanta and Sameep Mehta. Towards crafting text adversarial samples. *arXiv preprint arXiv:1707.02812*, 2017.

[Sato *et al.*, 2018] Motoki Sato, Jun Suzuki, Hiroyuki Shindo, and Yuji Matsumoto. Interpretable adversarial perturbation in input embedding space for text. In *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI 2018, July 13-19, 2018, Stockholm, Sweden*, pages 4323–4330, 2018.

[Sun *et al.*, 2016] Maosong Sun, Xinxiong Chen, Kaixu Zhang, Zhipeng Guo, and Zhiyuan Liu. Thulac: An efficient lexical analyzer for chinese. Technical report, Technical Report. Technical Report, 2016.

[Tiantian, 2015] Le Tiantian. Pinyin2hanzi, 2015.

[Wu *et al.*, 2019] Wei Wu, Yuxian Meng, Qinghong Han, Muyu Li, Xiaoya Li, Jie Mei, Ping Nie, Xiaofei Sun, and Jiwei Li. Glyce: Glyph-vectors for chinese character representations. *arXiv preprint arXiv:1901.10125*, 2019.

[Zhang *et al.*, 2015] Xiang Zhang, Junbo Zhao, and Yann LeCun. Character-level convolutional networks for text classification. In *Advances in neural information processing systems*, pages 649–657, 2015.

[Zhu *et al.*, 2018] Wenhao Zhu, Xin Jin, Jianyue Ni, Baogang Wei, and Zhiguo Lu. Improve word embedding using both writing and pronunciation. *PloS one*, 13(12):e0208785, 2018.