Punctuation-level Attack: Single-shot and Single Punctuation Attack Can Fool Text Models

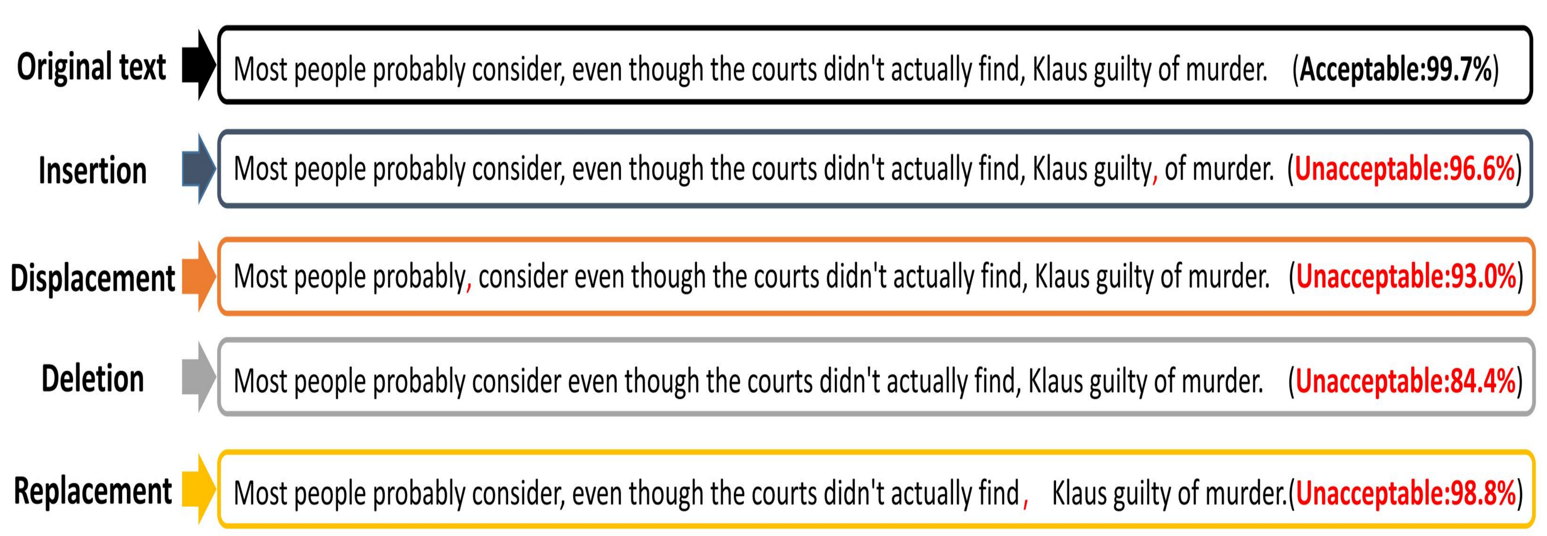
Wenqiang Wang[1] Chongyang Du[1] Tao Wang[2] Kaihao Zhang[3]
Wenhan Luo[1*] Lin Ma[4] Wei Liu[5] Xiaochun Cao[1]

[1]Shenzhen Campus of Sun Yat-sen University [2]Nanjing University
[3]Australian National University [4]Meituan [5]Tencent

Introduction

- Punctuation-level attacks: We first propose punctuation-level attacks, which regard the perturbations of punctuation as a systematic attack like character-level, word-level, and sentence-level attacks. We propose four primary modes of punctuation-level attacks and explain punctuation-level attacks from the perspective of optimal perturbations.
- TPPE: We first propose the TPPE embedding method to decrease the search cost. We reduce the query time complexity from $O\ (kn)$ of Insertion, $O\ (nt)$ of Displacement, $O\ (t)$ of Deletion, and $O\ (kt)$ of Replacement, to $O\ (1)$ under single punctuation attack. It can quickly and reasonably embed the adversarial candidate text xadv using a single-shot query.
- Single-shot and Single Punctuation Attack: To make our punctuation-level attack more imperceptible, we modify only one punctuation. Besides, we discuss single-punctuation attacks in the most challenging scenario: zero query, black-box function, hard-label output, one-punctuation limitation, and single-shot attack, which is the closest to the real-world scenarios. We correspondingly propose the TPPEP method and achieve promising experimental results.

Punctuation-level Attack



- Insertion: Punctuation p is inserted into the target text to fool the text model.
- Displacement: Punctuation p is moved from position i to position j in the target text.
- Deletion: Punctuation p is removed from the target text.
- Replacement: Punctuation p_i is replaced by p_j in the target text.

Embedding Method

We present the pseudo code for TPPE in this paper, using the Insertion mode as an example.

Algorithm 1 TPPE Embedding Method of `Insertion`

Input: The input text $\boldsymbol{x}$, the number of tokens n, the candidate punctuations p_i, the feature extraction function $f_{fe}(\boldsymbol{x})$
Output: the embedding of adversarial candidate text $\boldsymbol{x}_{adv}$
for $i = 1$ to n **do**
  $\boldsymbol{E}_{pos}^{i} = PE(i)$
end for
for $i = 1$ to k **do**
  $\boldsymbol{E}_{punc}^{i} = f_{fe}(\boldsymbol{p_i})$
end for
$\boldsymbol{E}_{text} = f_{fe}(\boldsymbol{x})$
for $i = 1$ to n **do**
  for $j = 1$ to k **do**
    $\boldsymbol{E}_{\boldsymbol{x}_{adv}^{ij}} = \boldsymbol{E}_{text} + \boldsymbol{E}_{posi} + \boldsymbol{E}_{punc}$
  end for
end for
$\boldsymbol{E}_{\boldsymbol{x}_{adv}} = \left[\boldsymbol{E}_{\boldsymbol{x}_{adv}^{11}}, \boldsymbol{E}_{\boldsymbol{x}_{adv}^{12}}, \ldots, \boldsymbol{E}_{\boldsymbol{x}_{adv}^{ik}}, \boldsymbol{E}_{\boldsymbol{x}_{adv}^{21}}, \ldots, \boldsymbol{E}_{\boldsymbol{x}_{adv}^{nk}}\right]$
return $\boldsymbol{E}_{\boldsymbol{x}_{adv}}$

According to Alg. 1, we reduce the query time complexity from $O\ (kn)$ of Insertion to $O\ (1)$ by using the TPPE method.

TPPEP Training Algorithm

Algorithm 2 TPPEP Training

Input: The training data $\boldsymbol{D} = \left\{\left(\boldsymbol{x}^1, \boldsymbol{x}_{adv}^1, y_{att}^1\right), \left(\boldsymbol{x}^2, \boldsymbol{x}_{adv}^2, y_{att}^2\right), \cdots, \left(\boldsymbol{x}^N, \boldsymbol{x}_{adv}^N, y_{att}^N\right)\right\}$. The $\boldsymbol{x}^i$ is input text, the $\boldsymbol{x}_{adv}^i$ is adversarial candidate text, and y_{att}^i is the result of attacking (successful attacking is denoted as label 1; else denoted as label 0). The max train epoch e_{max}, the substitute model f_{sub}, the embedding model $TPPE$
Output: The trained TPPEP model f_p
for $i = 1$ to N **do**
  $\boldsymbol{E}_{text}^{i} = f_{sub}(\boldsymbol{x}^i)$
  $\boldsymbol{E}_{\boldsymbol{x}_{adv}}^{i} = TPPE(\boldsymbol{x}_{adv}^i)$
  The input embedding $\boldsymbol{E}^i = concat(\boldsymbol{E}_{text}^i, \boldsymbol{E}_{\boldsymbol{x}_{adv}}^i)$
end for
The embedding of training data $\boldsymbol{ED} = \left\{\left(\boldsymbol{E}^1, y_{att}^1\right), \left(\boldsymbol{E}^2, y_{att}^1\right), \cdots, \left(\boldsymbol{E}^N, y_{att}^N\right)\right\}$
for $i = 1$ to e_{max} **do**
  // Train f_p on $\boldsymbol{ED}$ to adjust the parameters θ_{f_p}
  $\theta_{f_p} \leftarrow \mathrm{train}(f_p, \boldsymbol{ED})$
end for
$f_p = f_p(\boldsymbol{ED}; \theta_{f_p})$
return f_p

TPPEP Searching Algorithm

- After training the TPPEP model f_p, we consider all candidate adversarial texts $\boldsymbol{x}_{adv}$ of input text $\boldsymbol{x}$ and calculate the embedding $\boldsymbol{ED}$ of both $\boldsymbol{x}_{adv}$ and $\boldsymbol{x}$. We then apply the TPPEP method to $\boldsymbol{ED}$ and calculate the score of the successful attack. The adversarial candidate text with the highest paraphrasing score calculated by the TPPEP method is chosen to deploy the attack.

Experimental Results

- The results of Text classification task, paraphrase task, and natural language inference task.

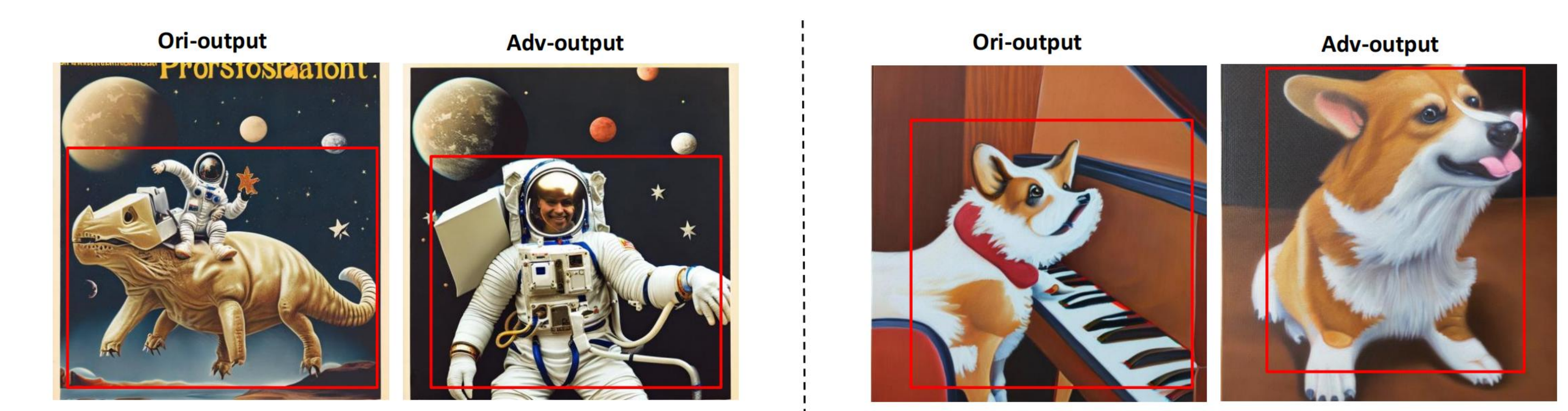
Cola	ELECTRA [7]					XLMR [37]					
mode	Top-1	Top-3	Top-5	Traversal	p_1	Top-1	Top-3	Top-5	Traversal	p_1	ASP
Insertion	67.40%	73.06%	73.83%	90.80%	74.23%	28.76%	52.64%	63.57%	93.67%	30.70%	362.62
Displacement	36.05%	66.35%	73.44%	80.44%	44.82%	43.05%	60.12%	76.03%	80.73%	53.33%	11.59
Deletion	5.18%	5.85%	5.94%	5.94%	87.21%	4.89%	5.85%	5.85%	5.85%	83.59%	1.15
Replacement	24.64%	36.82%	44.77%	74.59%	33.03%	6.62%	9.88%	12.37%	20.23%	32.72%	41.49
QQP	DistilBERT1 [36]					DistilBERT2 [36]					
Insertion	14.72%	18.76%	22.68%	47.18%	31.20%	8.67%	10.43%	11.73%	48.23%	17.98%	957.72
Displacement	8.52%	15.05%	18.86%	26.78%	31.81%	7.21%	12.43%	15.57%	23.44%	30.76%	36.57
Deletion	3.94%	5.93%	6.02%	6.03%	65.34%	5.06%	6.86%	6.95%	6.96%	72.70%	2.53
Replacement	7.59%	10.04%	12.18%	19.70%	38.53%	16.70%	20.97%	22.65%	29.65%	56.32%	90.91
Wanli	RoBERTa [27]					DeBERTa [14]					
Insertion	8.44%	19.22%	26.20%	66.74%	12.65%	15.28%	29.20%	37.40%	80.14%	19.07%	1161.12
Displacement	5.12%	9.14%	12.26%	26.14%	19.59%	10.28%	16.60%	20.34%	38.40%	26.77%	53.94
Deletion	3.22%	5.84%	6.14%	6.16%	52.27%	5.74%	8.58%	8.96%	8.98%	63.92%	2.94
Replacement	8.48%	15.96%	19.80%	45.82%	18.51%	6.92%	13.08%	16.88%	54.76%	12.64%	105.88

- The results of the semantic-similarity-scoring task

	Sentence-BERT		Distilbert	
STS12	Pearson	Spearman	Pearson	Spearman
Without Attack	0.7990	0.6988	0.8056	0.7257
TOP-1	0.7874	0.6862	0.7902	0.7035
TOP-3	0.7760	0.6738	0.7759	0.6990
TOP-5	0.7654	0.6626	0.7649	0.6745
Traversal	0.6992	0.5832	0.6994	0.6048

- The results of text to image and summarization task

Task	Metric	Without Attack	TOP-1	TOP-3	TOP-5	Traversal
Text to image	CLIP score	0.3278	0.3176	0.3069	0.3022	0.2610
Summarization	ROUGE-1	11.69	10.91	9.65	9.11	5.22



Ori-text: a professional photograph of an astronaut riding a triceratops
Adv-text: a professional photograph of an astronaut. riding a triceratops

Ori-text: a corgi is playing piano, oil on canvas
Adv-text: a corgi is playing, piano, oil on canvas

dataset	pokemon-blip-captions							
	all	train	test		Ori-image	Adv-image	Ori-text	Adv-text
Ori-text	0.3273	0.3272	0.3278	Ori-text0	0.3281	0.2484	1	0.9782
Adv-text	0.2591	0.2586	0.2610	Ori-text1	0.4040	0.3468	1	0.9843