Textual Adversarial Attack

Presenter: Junjie
Date: 2021-07-28

Background

- Adversarial Attack: $$\mathbf{x}' = \mathbf{x} + \eta, f(\mathbf{x}) = \mathbf{y}, \mathbf{x} \in \mathbf{X}$$ $$f(\mathbf{x}') \neq \mathbf{y}$$

- Adversarial attacks for discrete data is more challenging since it is difficult to directly adapt gradient-based methods.

- White box/Black box attack.

Textual Adversarial Attack

Classification

IMDB	Ori	i first seen this movie in the early 80s .. it really had nice picture quality too . anyways , i 'm glad i found this movie again ... the part i loved best was when he hijacked the car from this poor guy... this is a movie i could watch over and over again . i highly recommend it .	Positive
	Adv	i first seen this movie in the early 80s .. it really had nice picture quality too . anyways , i 'm glad i found this movie again ... the part i loved best was when he hijacked the car from this poor guy... this is a movie i could watch over and over again . i inordinately recommend it .	Negative

Generation

Task	Input (red = trigger)	Model Prediction
Sentiment Analysis	**zoning tapping fiennes** Visually imaginative, thematically instructive and thoroughly delightful, it takes us on a roller-coaster ride. . .	Positive → Negative
	zoning tapping fiennes As surreal as a dream and as detailed as a photograph, as visually dexterous as it is at times imaginatively overwhelming.	Positive → Negative

BERT-ATTACK: Adversarial Attack Against BERT Using BERT

Linyang Li, Ruotian Ma, Qipeng Guo, Xiangyang Xue, Xipeng Qiu*
Shanghai Key Laboratory of Intelligent Information Processing, Fudan University
School of Computer Science, Fudan University
825 Zhangheng Road, Shanghai, China
{linyangli19,rtma19,qpguo16,xyxue,xpqiu}@fudan.edu.cn

Algorithm 1 BERT-Attack

procedure WORD IMPORTANCE RANKING
  $S = [w_0, w_1, \cdots]$ // input: tokenized sentence
  $Y \leftarrow$ gold-label
  for w_i in S **do**
    calculate importance score I_{w_i} using Eq. 1
  select word list $L = [w_{top-1}, w_{top-2}, \cdots]$
  // sort S using I_{w_i} in descending order and collect $top - K$ words
procedure REPLACEMENT USING BERT
  $H = [h_0, \cdots, h_n]$ // sub-word tokenized sequence of S
  generate top-K candidates for all sub-words using BERT and get $P^{\in n \times K}$
  for w_j in L **do**
    if w_j is a whole word **then**
      get candidate $C = Filter(P^j)$
      replace word w_j
    else
      get candidate C using PPL ranking and Filter
      replace sub-words $[h_j, \cdots, h_{j+t}]$
    Find Possible Adversarial Sample
    for c_k in C **do**
      $S' = [w_0, \cdots, w_{j-1}, c_k, \cdots]$ // attempt
      if argmax$(o_y(S'))! = Y$ **then**
        return $S^{adv} = S'$ // success attack
      else
        if $o_y(S') < o_y(S^{adv})$ **then**
          $S^{adv} = [w_0, \cdots, w_{j-1}, c, \cdots]$ // do one perturbation
  return **None**

Word Importance Ranking

procedure WORD IMPORTANCE RANKING
  $S = [w_0, w_1, \cdots]$ // input: tokenized sentence
  $Y \leftarrow$ gold-label
  for w_i in S **do**
    calculate importance score I_{w_i} using Eq. 1
  select word list $L = [w_{top-1}, w_{top-2}, \cdots]$
  // sort S using I_{w_i} in descending order and collect $top - K$ words

Let $S = [w_0, \cdots, w_i \cdots]$ denote the input sentence, and $o_y(S)$ denote the logit output by the target model for correct label y, the importance score I_{w_i} is defined as

$$I_{w_i} = o_y(S) - o_y(S_{\backslash w_i}), \tag{1}$$

where $S_{\backslash w_i} = [w_0, \cdots, w_{i-1}, [\texttt{MASK}], w_{i+1}, \cdots]$ is the sentence after replacing w_i with $[\texttt{MASK}]$.

Replacement using BERT

- Input the whole sequence to MLM to generate candidates.
- Filtered stop words, sub-words and antonyms (sentiment analysis).

procedure Replacement using BERT
    $H = [h_0, \cdots, h_n]$ // sub-word tokenized sequence of S
    generate top-K candidates for all sub-words using BERT and get $P^{\in n \times K}$
    for w_j in L **do**
        if w_j is a whole word **then**
            get candidate $C = Filter(P^j)$
            replace word w_j
        else
            get candidate C using PPL ranking and Filter
            replace sub-words $[h_j, \cdots, h_{j+t}]$
        Find Possible Adversarial Sample
        for c_k in C **do**
            $S^{'} = [w_0, \cdots, w_{j-1}, c_k, \cdots]$ // attempt
            if $\text{argmax}(o_y(S^{'}))! = Y$ **then**
                return $S^{adv} = S^{'}$ // success attack
            else
                if $o_y(S^{'}) < o_y(S^{adv})$ **then**
                    $S^{adv} = [w_0, \cdots, w_{j-1}, c, \cdots]$ // do one perturbation
    return **None**

Datasets

Task	Dataset	Train	Test	Avg Len
Classification	AG's News	30K	1.9K	43
	Fake News	18.8K	2K	885
	MR	9K	1K	20
	IMDB	25K	25K	215
	Yelp	560K	38K	152
Entailment	SNLI	570K	3K	8
	MultiNLI	433K	10K	11

Table 1: Overview of the datasets.

Experiments

Dataset	Method	Original Acc	Attacked Acc	Perturb %	Query Number	Avg Len	Semantic Sim
Fake	BERT-Attack(ours)	97.8	**15.5**	**1.1**	**1558**	885	**0.81**
	TextFooler(Jin et al., 2019)		19.3	11.7	4403		0.76
	GA(Alzantot et al., 2018)		58.3	1.1	28508		-
Yelp	BERT-Attack(ours)	95.6	**5.1**	**4.1**	**273**	157	**0.77**
	TextFooler		6.6	12.8	743		0.74
	GA		31.0	10.1	6137		-
IMDB	BERT-Attack(ours)	90.9	**11.4**	**4.4**	**454**	215	**0.86**
	TextFooler		13.6	6.1	1134		**0.86**
	GA		45.7	4.9	6493		-
AG	BERT-Attack(ours)	94.2	**10.6**	**15.4**	**213**	43	**0.63**
	TextFooler		12.5	22.0	357		0.57
	GA		51	16.9	3495		-
SNLI	BERT-Attack(ours)	89.4(H/P)	7.4/**16.1**	**12.4/9.3**	**16/30**	8/18	0.40/**0.55**
	TextFooler		**4.0**/20.8	18.5/33.4	60/142		**0.45**/0.54
	GA		14.7/-	20.8/-	613/-		-
MNLI matched	BERT-Attack(ours)	85.1(H/P)	**7.9/11.9**	**8.8/7.9**	**19/44**	11/21	0.55/**0.68**
	TextFooler		9.6/25.3	15.2/26.5	78/152		**0.57**/0.65
	GA		21.8/-	18.2/-	692/-		-
MNLI mismatched	BERT-Attack(ours)	82.1(H/P)	**7/13.7**	**8.0/7.1**	**24/43**	12/22	0.53/**0.69**
	TextFooler		8.3/22.9	14.6/24.7	86/162		**0.58**/0.65
	GA		20.9/-	19.0/-	737/-		-

Gradient-based Adversarial Attacks against Text Transformers

Chuan Guo* **Alexandre Sablayrolles*** **Hervé Jégou** **Douwe Kiela**
Facebook AI Research

Core idea

$$(\tilde{\pi}_i)_j := \frac{\exp((\Theta_{i,j} + g_{i,j})/T)}{\sum_{v=1}^{V} \exp((\Theta_{i,v} + g_{i,v})/T)}, \qquad (7)$$

where $g_{i,j} \sim \mathrm{Gumbel}(0,1)$ and $T > 0$ is a temperature parameter that controls the smoothness of the Gumbel-softmax distribution. As $T \to 0$,

We can now optimize Θ using gradient descent by defining a smooth approximation of the objective function in Equation 5:

$$\min_{\Theta \in \mathbb{R}^{n \times V}} \mathbb{E}_{\tilde{\boldsymbol{\pi}} \sim \tilde{P}_\Theta} \ell(\mathbf{e}(\tilde{\boldsymbol{\pi}}), y; h), \qquad (8)$$

$$\begin{aligned} \mathcal{L}(\Theta) =& \mathbb{E}_{\tilde{\boldsymbol{\pi}} \sim \tilde{P}_\Theta} \ell(\mathbf{e}(\tilde{\boldsymbol{\pi}}), y; h) \\ & + \lambda_{\mathrm{lm}} \, \mathrm{NLL}_g(\tilde{\boldsymbol{\pi}}) + \lambda_{\mathrm{sim}} \, \rho_g(\mathbf{x}, \tilde{\boldsymbol{\pi}}), \end{aligned} \qquad (10)$$

White Box Results

Task	GPT-2			XLM (en-de)			BERT		
	Clean Acc.	Adv. Acc.	Cosine Sim.	Clean Acc.	Adv. Acc.	Cosine Sim.	Clean Acc.	Adv. Acc.	Cosine Sim.
DBPedia	99.2	5.2	0.91	99.1	7.6	0.80	99.2	7.1	0.80
AG News	94.8	6.6	0.90	94.4	5.4	0.87	95.1	2.5	0.82
Yelp	97.8	2.9	0.94	96.3	3.4	0.93	97.3	4.7	0.92
IMDB	93.8	7.6	0.98	87.6	0.1	0.97	93.0	3.0	0.92
MNLI (m.)	81.7	2.8/11.0	0.82/0.88	76.9	1.3/8.4	0.74/0.80	84.6	7.1/10.2	0.87/0.92
MNLI (mm.)	82.5	4.2/13.5	0.85/0.88	76.3	1.3/8.9	0.75/0.80	84.5	7.4/8.8	0.89/0.93

Table 1: Result of white-box attack against three transformer models: GPT-2, XLM (en-de), and BERT. Our attack is able to reduce the target model's accuracy to below 10% in almost all cases, while maintaining a high level of semantic similarity (cosine similarity of higher than 0.8 using USE embeddings).

Transfer to Black-Box Scheme

- Workflow:

1. Use a GPT-2 and 1000 examples to train adversarial distributions.
2. Use these 1000 distributions and gumbel-softmax to sample adversarial examples for other black-box models.

Task	Clean Acc.	Attack Alg.	Adv. Acc.	# Queries	Cosine Sim.
AG News	95.1	GBDA (ours)	**8.8**	**107**	0.69
		BERT-Attack	10.6	213	0.63
		BAE	13.0	419	**0.75**
		TextFooler	12.6	357	0.57
Yelp	97.3	GBDA (ours)	**2.6**	**43**	0.83
		BERT-Attack	5.1	273	0.77
		BAE	12.0	434	**0.90**
		TextFooler	6.6	743	0.74
IMDB	93.0	GBDA (ours)	**8.5**	**116**	0.92
		BERT-Attack	11.4	454	0.86
		BAE	24.0	592	**0.95**
		TextFooler	13.6	1134	0.86
MNLI (m.)	84.6	GBDA (ours)	**2.3/10.8**	37/133	0.75/0.79
		BERT-Attack	7.9/11.9	**19/44**	0.55/0.68
		BAE	25.4/36.2	68/120	**0.88/0.88**
		TextFooler	9.6/25.3	78/152	0.57/0.65
MNLI (mm.)	84.5	GBDA (ours)	**1.8/13.4**	30/159	0.76/0.80
		BERT-Attack	7/13.7	**24/43**	0.53/0.69
		BAE	19.2/30.3	75/110	**0.88/0.88**
		TextFooler	8.3/22.9	86/162	0.58/0.65

Table 3: Evaluation of black-box transfer attack from GPT-2 to finetuned BERT classifiers. Our attack is able exceed the attack performance of BERT-Attack and BAE, while maintaining a higher semantic similarity with fewer number of queries in most cases. Furthermore, our transfer attack does not require continuous-valued outputs, which all the baseline methods rely on.

Target Model	Task	Clean Acc.	Adv. Acc.	# Queries	Cosine Sim.
ALBERT	**AG News**	94.7	7.5	84	0.68
	Yelp	97.5	5.9	76	0.79
	IMDB	93.8	13.1	157	0.87
RoBERTA	**AG News**	94.7	10.7	130	0.67
	IMDB	95.2	17.4	205	0.87
	MNLI (m.)	88.1	4.1/15.1	63/179	0.69/0.76
	MNLI (mm.)	87.8	3.2/15.9	51/189	0.69/0.78
XLNet	**IMDB**	93.8	12.1	149	0.87
	MNLI (m.)	87.2	3.9/13.7	56/162	0.70/0.77
	MNLI (mm.)	86.8	1.7/14.4	32/171	0.70/0.78

Table 4: Result of black-box transfer attack from GPT-2 to other transformer models. Our attack is achieved by sampling from the same adversarial distribution P_Θ and is able to generalize to the three target transformer models considered in this study.

Seq2Sick: Evaluating the Robustness of Sequence-to-Sequence Models with Adversarial Examples

Minhao Cheng,[1] Jinfeng Yi,[2] Pin-Yu Chen,[3] Huan Zhang,[1] Cho-Jui Hsieh[1]
[1]Department of Computer Science, UCLA, [2]JD AI Research,[3]IBM Research
{mhcheng, huanzhang, chohsieh}@cs.ucla.edu, yijinfeng@jd.com, pin-yu.chen@ibm.com

Two attack settings

- Non-overlapping attack
  - This attack requires that the output of the adversarial example shares no overlapping words with the original output.
- Targeted keywords attack
  - Given a set of targeted keywords, the goal of targeted keywords attack is to find an adversarial input sequence such that all the keywords must appear in its corresponding output.

Key ideas

Algorithm 1 Seq2Sick algorithm

Input: input sequence $\mathbf{x} = \{x_1, \ldots, x_N\}$, seq2seq model, target keyword $\{k_1, \ldots, k_T\}$
Output: adversarial sequence $\mathbf{x}^* = \mathbf{x} + \boldsymbol{\delta}^*$
Let $\mathbf{s} = \{s_1, \ldots, s_M\}$ denote the original output of $\mathbf{x}$.
Set the loss $L(\cdot)$ in (9) to be (3)
if Targeted Keyword Attack **then**
  Set the loss $L(\cdot)$ in (9) to be (7)
end if
for $r = 1, 2, \ldots, T$ **do**
  back-propagation L to achieve gradient $\nabla_{\boldsymbol{\delta}} L(\mathbf{x} + \boldsymbol{\delta}_r)$
  for $i = 1, 2, \ldots, N$ **do**
    $\delta_{r,i} = 0$
    if $\|\delta_{r,i}\| > \eta\lambda_1$ **then**
      $\delta_{r,i} = \delta_{r,i} - \eta\lambda_1 \frac{\delta_{r,i}}{\|\delta_{r,i}\|}$
    end if
  end for
  $y^{r+1} = \boldsymbol{\delta}^r + \eta \cdot \nabla_{\boldsymbol{\delta}} L(\mathbf{x} + \boldsymbol{\delta}^r)$
  $\boldsymbol{\delta}^{r+1} = \underset{\mathbf{x}+\boldsymbol{\delta}^{r+1} \in \mathbb{W}}{\arg\min} \left\| y^{r+1} - \boldsymbol{\delta}^{r+1} \right\|$
end for
$\boldsymbol{\delta}^* = \boldsymbol{\delta}^T$
$\mathbf{x}^* = \mathbf{x} + \boldsymbol{\delta}^*$
return $\mathbf{x}^*$

$$\min_{\boldsymbol{\delta}} L(\mathbf{X}+\boldsymbol{\delta}) + \lambda_1 \sum_{i=1}^{N} \|\delta_i\|_2 + \lambda_2 \sum_{i=1}^{N} \min_{\mathbf{w}_j \in \mathbb{W}} \{\|\mathbf{x}_i + \delta_i - \mathbf{w}_j\|_2\}$$
$$\text{s.t. } \mathbf{x}_i + \delta_i \in \mathbb{W} \quad \forall i = 1, \ldots, N \tag{9}$$

$$L_{\text{non-overlapping}} = \sum_{t=1}^{M} \max\{-\epsilon,\ z_t^{(s_t)} - \max_{y \neq s_t}\{z_t^{(y)}\}\}, \tag{3}$$

$$\sum_{i=1}^{|K|} \min_{t \in [M]} \{m_t(\max\{-\epsilon,\ \max_{y \neq k_i}\{z_t^{(y)}\} - z_t^{(k_i)}\})\}. \tag{7}$$

Datasets

Table 2: Statistics of the datasets. "# Samples" is the number of test examples we used for robustness evaluations

DATASETS	# SAMPLES	AVERAGE INPUT LENGTHS
GIGAWORD	1,000	30.1 WORDS
DUC2003	624	35.5 WORDS
DUC2004	500	35.6 WORDS
MULTI30K	500	11.5 WORDS

Experiments

Table 3: Results of non-overlapping attack in text summarization. **# changed** is how many words are changed in the input sentence. The high BLEU scores and low average number of changed words indicate that the crafted adversarial inputs are very similar to their originals, and we achieve high success rates to generate a summarization that differs with the original *at every position* for all three datasets.

Dataset	Success%	BLEU	# changed
Gigaword	86.0%	0.828	2.17
DUC2003	85.2%	0.774	2.90
DUC2004	84.2%	0.816	2.50

Table 4: Results of targeted keywords attack in text summarization. $|K|$ is the number of keywords. We found that our method can make the summarization include 1 or 2 target keywords with a high success rate, while the changes made to the input sentences are relatively small, as indicated by the high BLEU scores and low average number of changed words. When $|K| = 3$, this task becomes more challenging, but our algorithm can still find many adversarial examples.

Datasest	$\|K\|$	Success%	BLEU	# changed
Gigaword	1	99.8%	0.801	2.04
	2	96.5%	0.523	4.96
	3	43.0%	0.413	8.86
DUC2003	1	99.6%	0.782	2.25
	2	87.6%	0.457	5.57
	3	38.3%	0.376	9.35
DUC2004	1	99.6%	0.773	2.21
	2	87.8%	0.421	5.1
	3	37.4%	0.340	9.3

Experiments

Table 5: Results of non-overlapping method and targeted keywords method in machine translation.

Method	Success%	BLEU	# changed
Non-overlap	89.4%	0.349	3.5
1-keyword	100.0%	0.705	1.8
2-keyword	91.0 %	0.303	4.0
3-keyword	69.6%	0.205	5.3

My Research: A Targeted Attack for Sequential Models

- Given:
  - Input sequence $x = (x_1, x_2, \ldots.., x_n)$
  - Output sequence $y = (y_1, y_2, \ldots.., y_k)$(n = k for sequence tagging tasks).
  - Black box model M that only outputs:
    - Logit distribution for each position in y (NMT) / **only the targeted words (hard label attack).**
  - Named entity list with corresponding translations/tags.

- Our Goal: we build an adversarial sequence x' and generate y':
  - In NER, at the specific positions of y', the attacked tags are different from the original tags.
  - In NMT, none of the translated tokens of the given NE appears in y'.
  - If an error appears in one of the entities, we say that we attack this sentence successfully.
  - x' is similar to x, measured by some proposed metrics.