

꼼꼼한 딥러닝 논문 리뷰와 코드 실습

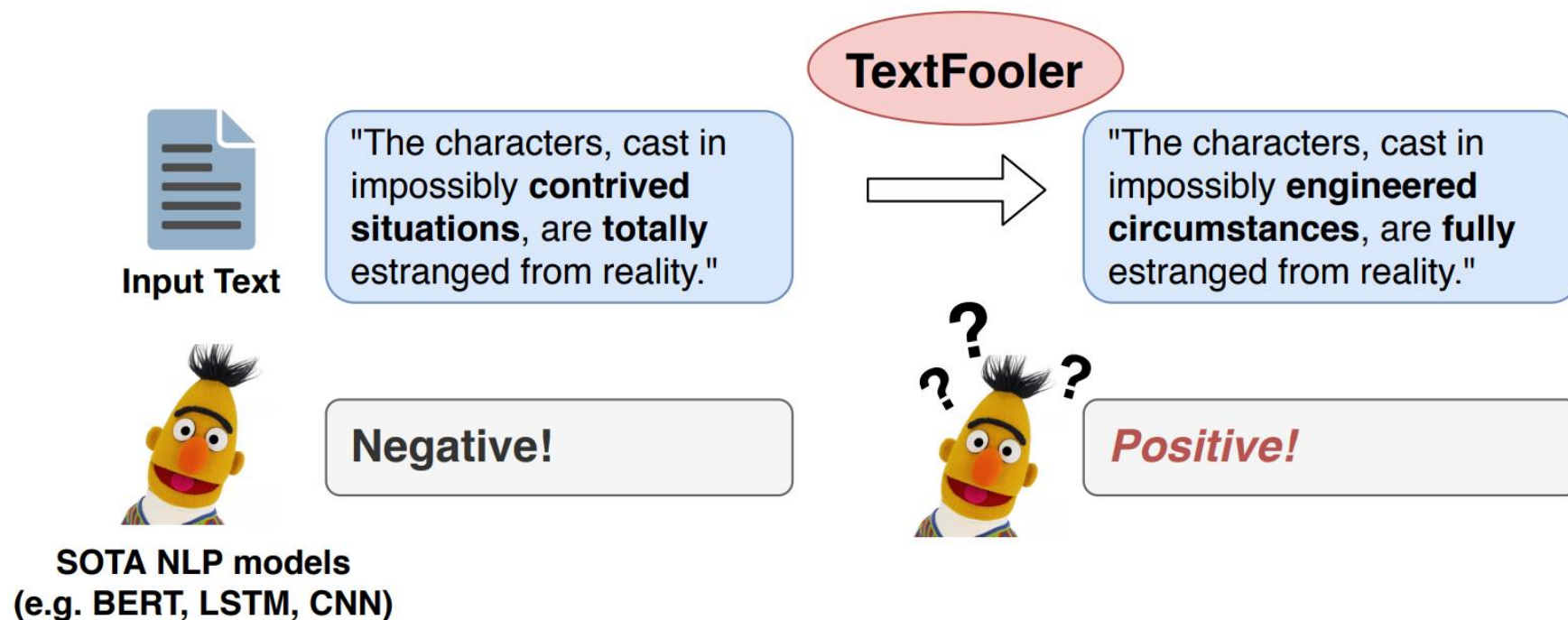
Deep Learning Paper Review and Code Practice

나동빈(dongbinna@postech.ac.kr)

Pohang University of Science and Technology

TextFooler (AAAI 2020)


- 본 논문은 텍스트 분류 모델을 속일 수 있는 강력한 공격 기법인 TextFooler를 제안합니다.
- 원본 텍스트를 약간 변경하여 **영화 리뷰(movie review)** 모델을 속일 수 있습니다.
 - 분류 모델 공격 예시를 확인해 봅시다.



텍스트 분야의 적대적 공격(Adversarial Attack)이 어려운 이유

- 이미지 도메인
 - 픽셀의 값들이 연속적(continuous)입니다.
 - 많은 픽셀에 조금씩 노이즈를 섞어도 인간의 눈에 잘 띄지 않습니다.
- 자연어 도메인
 - 단어(word)나 문자(character)가 불연속적인 토큰입니다.
 - 약간의 변화가 발생해도 인간의 눈에 잘 띄는 편입니다.

40	40	95	50
50	30	50	90
100	40	40	100
50	60	80	140



Attack

42	38	97	48
48	32	52	92
98	38	42	102
48	58	82	142

[Figure] 이미지 도메인에서의 공격 결과 예시

“I love you so much”  “I love you **a lot**”

Attack

[Figure] 자연어 도메인에서의 공격 결과 예시

본 논문에서는 어떤 일을 했을까요?

- Proposing *TextFooler*
 1. Effective: Outperforming previous attacks by success rate and perturbation rate.
 2. Utility-preserving: Preserving semantic content, grammaticality, human prediction.
 3. Efficient: Generating adversarial text with low computational complexity.
- Extensive experiments
 - Models: Three state-of-the-art deep learning models.
 - Datasets: Five text classification tasks, two textual entailment tasks.

Textual Entailment 문제 소개

- Textual entailment denotes a directional relation between sentences.
- Hypothesis: If you help the needy, God will reward you.
- Examples
 - Entailment: Giving money to a poor man has good consequences.
 - Contradiction: Giving money to a poor man has no consequences.
 - Neutral relationship: Giving money to a poor man will make you a better person.

본 논문이 해결하고자 하는 문제는 무엇일까요?

- The adversarial example should conform to the following requirements:

$$F(X_{\text{adv}}) \neq F(X), \text{ and } \text{Sim}(X_{\text{adv}}, X) \geq \epsilon$$

- $\text{Sim}(\cdot)$ is often a semantic and syntactic similarity function.
- $F(\cdot)$ is a targeted model.

본 논문이 제안한 공격 기법의 유형

- Black-box attack
 - Querying the target model, getting as results the predictions and confidence scores.
- Targeted & Non-targeted attack
 - Fooling the model to make an inaccurate prediction.
- Word-wise perturbing attack
 - Changing word by word.

본 논문의 공격 알고리즘 소개

1. Word Importance Ranking

2. Word Transformer

- 1) Synonym extraction: Finding similar words.
- 2) POS checking: Keeping the words with the same part-of-speech (POS).
- 3) Semantic similarity checking: Preserving semantic meaning of the perturbed sentence.
- 4) Finalization of adversarial examples: Finding any candidate that can alter the prediction.

1. Word Importance Ranking (단어 중요도 순위 매기기)

- 주어진 문장이 n 개의 단어 $X = \{w_1, w_2, \dots, w_n\}$ 로 구성될 때, 일반적으로 오직 몇 개의 단어만 실제 분류 결과에 큰 영향을 미칩니다.
- 따라서 중요한 단어 위주로 변경한다면 공격이 더욱 잘 수행될 것입니다.

Algorithm 1 Adversarial Attack by TEXTFOOLER

Input: Sentence example $X = \{w_1, w_2, \dots, w_n\}$, the corresponding ground truth label Y , target model F , sentence similarity function $\text{Sim}(\cdot)$, sentence similarity threshold ϵ , word embeddings Emb over the vocabulary Vocab .

Output: Adversarial example X_{adv}

- 1: Initialization: $X_{\text{adv}} \leftarrow X$
- 2: **for** each word w_i in X **do**
- 3: Compute the importance score I_{w_i} via Eq. (2)
- 4: **end for**
- 5:
- 6: Create a set W of all words $w_i \in X$ sorted by the descending order of their importance score I_{w_i} .

1. Word Importance Ranking (단어 중요도 순위 매기기)

- We can calculate the **Importance score** I_{w_i} as the prediction change before and after deleting the word w_i , which is formally defined as follows:

$$I_{w_i} = \begin{cases} F_Y(X) - F_Y(X_{\setminus w_i}), & \text{if } F(X) = F(X_{\setminus w_i}) = Y \\ (F_Y(X) - F_Y(X_{\setminus w_i})) + (F_{\bar{Y}}(X_{\setminus w_i}) - F_{\bar{Y}}(X)), & \text{if } F(X) = Y, F(X_{\setminus w_i}) = \bar{Y}, \text{ and } Y \neq \bar{Y}. \end{cases}$$

- $X_{\setminus w_i} = X \setminus \{w_i\} = \{w_1, \dots, w_{i-1}, w_{i+1}, \dots, w_n\}$

2-1) Synonym Extraction (동의어 추출)

```
7: Filter out the stop words in  $W$ .
8: for each word  $w_j$  in  $W$  do
9:   Initiate the set of candidates CANDIDATES by extracting
   the top  $N$  synonyms using  $\text{CosSim}(\text{Emb}_{w_j}, \text{Emb}_{\text{word}})$  for
   each word in Vocab.
10:  CANDIDATES  $\leftarrow$  POSFilter(CANDIDATES)
11:  FINCANDIDATES  $\leftarrow \{ \}$ 
12:  for  $c_k$  in CANDIDATES do
13:     $X' \leftarrow$  Replace  $w_j$  with  $c_k$  in  $X_{\text{adv}}$ 
14:    if  $\text{Sim}(X', X_{\text{adv}}) > \epsilon$  then
15:      Add  $c_k$  to the set FINCANDIDATES
16:       $Y_k \leftarrow F(X')$ 
17:       $P_k \leftarrow F_{Y_k}(X')$ 
18:    end if
19:  end for
20:  if there exists  $c_k$  whose prediction result  $Y_k \neq Y$  then
21:    In FINCANDIDATES, only keep the candidates  $c_k$  whose
    prediction result  $Y_k \neq Y$ 
22:     $c^* \leftarrow \underset{c \in \text{FINCANDIDATES}}{\text{argmax}} \text{Sim}(X, X'_{w_j \rightarrow c})$ 
23:     $X_{\text{adv}} \leftarrow$  Replace  $w_j$  with  $c^*$  in  $X_{\text{adv}}$ 
24:    return  $X_{\text{adv}}$ 
25:  else if  $P_{Y_k}(X_{\text{adv}}) > \min_{c_k \in \text{FINCANDIDATES}} P_k$  then
26:     $c^* \leftarrow \underset{c_k \in \text{FINCANDIDATES}}{\text{argmin}} P_k$ 
27:     $X_{\text{adv}} \leftarrow$  Replace  $w_j$  with  $c^*$  in  $X_{\text{adv}}$ 
28:  end if
29: end for
30: return None
```

- CANDIDATES (후보): N closest synonyms according to the cosine similarity between w_i and every other word in the vocabulary.
- Using the attacker's word embedding model.

2-2) POS Checking (품사 체크하기)

```
7: Filter out the stop words in  $W$ .
8: for each word  $w_j$  in  $W$  do
9:   Initiate the set of candidates CANDIDATES by extracting
   the top  $N$  synonyms using  $\text{CosSim}(\text{Emb}_{w_j}, \text{Emb}_{\text{word}})$  for
   each word in Vocab.
10:  CANDIDATES  $\leftarrow$  POSFilter(CANDIDATES)
11:  FINCANDIDATES  $\leftarrow \{ \}$ 
12:  for  $c_k$  in CANDIDATES do
13:     $X' \leftarrow$  Replace  $w_j$  with  $c_k$  in  $X_{\text{adv}}$ 
14:    if  $\text{Sim}(X', X_{\text{adv}}) > \epsilon$  then
15:      Add  $c_k$  to the set FINCANDIDATES
16:       $Y_k \leftarrow F(X')$ 
17:       $P_k \leftarrow F_{Y_k}(X')$ 
18:    end if
19:  end for
20:  if there exists  $c_k$  whose prediction result  $Y_k \neq Y$  then
21:    In FINCANDIDATES, only keep the candidates  $c_k$  whose
    prediction result  $Y_k \neq Y$ 
22:     $c^* \leftarrow \underset{c \in \text{FINCANDIDATES}}{\text{argmax}} \text{Sim}(X, X'_{w_j \rightarrow c})$ 
23:     $X_{\text{adv}} \leftarrow$  Replace  $w_j$  with  $c^*$  in  $X_{\text{adv}}$ 
24:    return  $X_{\text{adv}}$ 
25:  else if  $P_{Y_k}(X_{\text{adv}}) > \min_{c_k \in \text{FINCANDIDATES}} P_k$  then
26:     $c^* \leftarrow \underset{c_k \in \text{FINCANDIDATES}}{\text{argmin}} P_k$ 
27:     $X_{\text{adv}} \leftarrow$  Replace  $w_j$  with  $c^*$  in  $X_{\text{adv}}$ 
28:  end if
29: end for
30: return None
```



- In the set CANDIDATES(후보) of the word w_i , we only keep the words with the same POS(품사) as w_i .
- 단어가 바뀐 뒤에도 품사가 바뀌지 않도록 함으로써 원본 문장과 문법적으로 다르지 않도록 합니다.

2-3) Semantic Similarity Checking (문맥 유사도 체크하기)

```
7: Filter out the stop words in  $W$ .
8: for each word  $w_j$  in  $W$  do
9:   Initiate the set of candidates  $CANDIDATES$  by extracting
   the top  $N$  synonyms using  $\text{CosSim}(\text{Emb}_{w_j}, \text{Emb}_{\text{word}})$  for
   each word in Vocab.
10:   $CANDIDATES \leftarrow \text{POSFilter}(CANDIDATES)$ 
11:   $\text{FINCANDIDATES} \leftarrow \{ \}$ 
12:  for  $c_k$  in  $CANDIDATES$  do
13:     $X' \leftarrow \text{Replace } w_j \text{ with } c_k \text{ in } X_{\text{adv}}$ 
14:    if  $\text{Sim}(X', X_{\text{adv}}) > \epsilon$  then
15:      Add  $c_k$  to the set  $\text{FINCANDIDATES}$ 
16:       $Y_k \leftarrow F(X')$ 
17:       $P_k \leftarrow F_{Y_k}(X')$ 
18:    end if
19:  end for
20:  if there exists  $c_k$  whose prediction result  $Y_k \neq Y$  then
21:    In  $\text{FINCANDIDATES}$ , only keep the candidates  $c_k$  whose
    prediction result  $Y_k \neq Y$ 
22:     $c^* \leftarrow \underset{c \in \text{FINCANDIDATES}}{\text{argmax}} \text{Sim}(X, X'_{w_j \rightarrow c})$ 
23:     $X_{\text{adv}} \leftarrow \text{Replace } w_j \text{ with } c^* \text{ in } X_{\text{adv}}$ 
24:    return  $X_{\text{adv}}$ 
25:  else if  $P_{Y_k}(X_{\text{adv}}) > \min_{c_k \in \text{FINCANDIDATES}} P_k$  then
26:     $c^* \leftarrow \underset{c_k \in \text{FINCANDIDATES}}{\text{argmin}} P_k$ 
27:     $X_{\text{adv}} \leftarrow \text{Replace } w_j \text{ with } c^* \text{ in } X_{\text{adv}}$ 
28:  end if
29: end for
30: return None
```

- $c \in CANDIDATES$
- $X_{\text{adv}} = \{w_i, \dots, w_{i-1}, c, w_{i+1}, \dots, w_n\}$
- Calculating semantic similarity between X and X_{adv} .
- Saving the prediction score $F(X_{\text{adv}})$.



2-4) Finalization of Adversarial Examples (결과적으로 단어를 변경해 공격하기)

```
7: Filter out the stop words in  $W$ .
8: for each word  $w_j$  in  $W$  do
9:   Initiate the set of candidates CANDIDATES by extracting
   the top  $N$  synonyms using  $\text{CosSim}(\text{Emb}_{w_j}, \text{Emb}_{\text{word}})$  for
   each word in Vocab.
10:  CANDIDATES  $\leftarrow$  POSFilter(CANDIDATES)
11:  FINCANDIDATES  $\leftarrow \{ \}$ 
12:  for  $c_k$  in CANDIDATES do
13:     $X' \leftarrow$  Replace  $w_j$  with  $c_k$  in  $X_{\text{adv}}$ 
14:    if  $\text{Sim}(X', X_{\text{adv}}) > \epsilon$  then
15:      Add  $c_k$  to the set FINCANDIDATES
16:       $Y_k \leftarrow F(X')$ 
17:       $P_k \leftarrow F_{Y_k}(X')$ 
18:    end if
19:  end for
20:  if there exists  $c_k$  whose prediction result  $Y_k \neq Y$  then
21:    In FINCANDIDATES, only keep the candidates  $c_k$  whose
    prediction result  $Y_k \neq Y$ 
22:     $c^* \leftarrow \underset{c \in \text{FINCANDIDATES}}{\text{argmax}} \text{Sim}(X, X'_{w_j \rightarrow c})$ 
23:     $X_{\text{adv}} \leftarrow$  Replace  $w_j$  with  $c^*$  in  $X_{\text{adv}}$ 
24:    return  $X_{\text{adv}}$ 
25:  else if  $P_{Y_k}(X_{\text{adv}}) > \min_{c_k \in \text{FINCANDIDATES}} P_k$  then
26:     $c^* \leftarrow \underset{c_k \in \text{FINCANDIDATES}}{\text{argmin}} P_k$ 
27:     $X_{\text{adv}} \leftarrow$  Replace  $w_j$  with  $c^*$  in  $X_{\text{adv}}$ 
28:  end if
29: end for
30: return None
```

- If there exists any candidate that can already alter the prediction of the target model, then we choose the word with the highest semantic similarity score among these winning candidates.
- If not, then we select the word with the least confidence score of label y as the best replacement word for w_i .
- And repeat Step 2 to transform the next selected word.

Datasets

- 공격을 위해 사용한 데이터 셋은 다음과 같습니다.
 - 5개의 text classification tasks, 2개의 textual entailment tasks.

Task	Dataset	Train	Test	Average Length	# of Output Class
Classification	AG's News	120K	7.6K	43	4
	Fake News	18.8K	2K	885	2
	MR	9K	1K	20	2
	IMDB	25K	25K	215	2
	Yelp	560K	38K	152	2
Entailment	SNLI	570K	3K	8	3
	MultiNLI	433K	10K	11	3

[Table] Overview of the datasets.

공격 대상 모델(Attack Target Models)

- Targeted models: For each dataset, training three state-of-the-art models.

	WordCNN	WordLSTM	BERT
AG	92.5	93.1	94.6
Fake	99.9	99.9	99.9
MR	79.9	82.2	85.8
IMDB	89.7	91.2	92.2
Yelp	95.2	96.6	96.1
	InferSent	ESIM	BERT
SNLI	84.6	88.0	90.7
MultiNLI	71.1/71.5	76.9/76.5	83.9/84.1

[Table] Original accuracy of target models on standard test sets.

공격 결과(Attack Results) 요약

	WordCNN					WordLSTM					BERT				
	MR	IMDB	Yelp	AG	Fake	MR	IMDB	Yelp	AG	Fake	MR	IMDB	Yelp	AG	Fake
Original Accuracy	78.0	89.2	93.8	91.5	96.7	80.7	89.8	96.0	91.3	94.0	86.0	90.9	97.0	94.2	97.8
After-Attack Accuracy	2.8	0.0	1.1	1.5	15.9	3.1	0.3	2.1	3.8	16.4	11.5	13.6	6.6	12.5	19.3
% Perturbed Words	14.3	3.5	8.3	15.2	11.0	14.9	5.1	10.6	18.6	10.1	16.7	6.1	13.9	22.0	11.7
Semantic Similarity	0.68	0.89	0.82	0.76	0.82	0.67	0.87	0.79	0.63	0.80	0.65	0.86	0.74	0.57	0.76
Query Number	123	524	487	228	3367	126	666	629	273	3343	166	1134	827	357	4403
Average Text Length	20	215	152	43	885	20	215	152	43	885	20	215	152	43	885

[Table] Automatic evaluation results of the attack system on text classification datasets.

	InferSent		ESIM		BERT	
	SNLI	MultiNLI (m/mm)	SNLI	MultiNLI (m/mm)	SNLI	MultiNLI (m/mm)
Original Accuracy	84.3	70.9/69.6	86.5	77.6/75.8	89.4	85.1/82.1
After-Attack Accuracy	3.5	6.7/6.9	5.1	7.7/7.3	4.0	9.6/8.3
% Perturbed Words	18.0	13.8/14.6	18.1	14.5/14.6	18.5	15.2/14.6
Semantic Similarity	0.50	0.61/0.59	0.47	0.59/0.59	0.45	0.57/0.58
Query Number	57	70/83	58	72/87	60	78/86
Average Text Length	8	11/12	8	11/12	8	11/12

[Table] Automatic evaluation results of the attack system on textual entailment datasets.

적대적 예제(Adversarial Attack Examples) 살펴보기

Movie Review (Positive (POS) ↔ Negative (NEG))	
Original (Label: NEG)	The characters, cast in impossibly <i>contrived situations</i> , are <i>totally</i> estranged from reality.
Attack (Label: POS)	The characters, cast in impossibly <i>engineered circumstances</i> , are <i>fully</i> estranged from reality.
Original (Label: POS)	It cuts to the <i>knot</i> of what it actually means to face your <i>scares</i> , and to ride the <i>overwhelming metaphorical wave</i> that life wherever it takes you.
Attack (Label: NEG)	It cuts to the <i>core</i> of what it actually means to face your <i>fears</i> , and to ride the <i>big metaphorical wave</i> that life wherever it takes you.
SNLI (Entailment (ENT), Neutral (NEU), Contradiction (CON))	
Premise	Two small boys in blue soccer uniforms use a wooden set of steps to wash their hands.
Original (Label: CON)	The boys are in band <i>uniforms</i> .
Adversary (Label: ENT)	The boys are in band <i>garment</i> .
Premise	A child with wet hair is holding a butterfly decorated beach ball.
Original (Label: NEU)	The <i>child</i> is at the <i>beach</i> .
Adversary (Label: ENT)	The <i>youngster</i> is at the <i>shore</i> .

[Table] Examples of original and adversarial sentences from MR (WordLSTM) and SNLI (BERT) datasets.

기존에 제안된 다른 공격 기법들과 결과 비교하기

Dataset	Model	Success Rate	% Perturbed Words
IMDB	(Li et al. 2018)	86.7	6.9
	(Alzantot et al. 2018)	97.0	14.7
	Ours	99.7	5.1
SNLI	(Alzantot et al. 2018)	70.0	23.0
	Ours	95.8	18.0
Yelp	(Kuleshov et al. 2018)	74.8	-
	Ours	97.8	10.6

- “Generating Natural Language Adversarial Examples”, Alzantot et al. [EMNLP 2018]:
 - Using *genetic algorithms*
 1. Randomly select a word in the sentence.
 2. Select a suitable replacement word that has a similar semantic meaning.
 3. Increase the target label prediction score by finding the best replacement.
 - No gradient required (black-box)

추가적인 실험 결과 분석하기

- Transferability

		WordCNN	WordLSTM	BERT
IMDB	WordCNN	—	84.9	90.2
	WordLSTM	74.9	—	87.9
	BERT	84.1	85.1	—
		InferSent	ESIM	BERT
SNLI	InferSent	—	62.7	67.7
	ESIM	49.4	—	59.3
	BERT	58.2	54.6	—

[Table] Transferability of adversarial examples on IMDB and SNLI dataset.

- Adversarial Training

	MR		SNLI	
	Af. Acc.	Pert.	Af. Acc.	Pert.
Original	11.5	16.7	4.0	18.5
+ Adv. Training	18.7	21.0	8.3	20.1

[Table] Comparison of after-attack accuracy and percentage of perturbed words of original training and adversarial training.

결론(Conclusion)

- Authors propose an efficient method, *TextFooler*, generating targeted adversarial texts.
- *TextFooler* can successfully attack state-of-the-art text classification and textual entailment models under the black-box setting.