Black-box Generation of Adversarial Text Sequences to Evade Deep Learning Classifiers

Ji Gao[1], Jack Lanchantin[1], Mary Lou Soffa[1], Yanjun Qi[1]

[1]University of Virginia
http://trustworthymachinelearning.org/

@ 1st Deep Learning and Security Workshop ; 2018

Outline

1. Motivation
   - White box vs. black box
2. Method
   - Word scorer
   - Word transformer
3. Experiment
4. Conclusions

Example of black-box classification systems

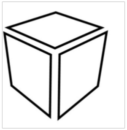
Google Perspective API

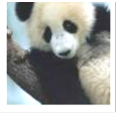
Example of black-box classification systems

Google Perspective API

Target scenario

Previous Research





Image

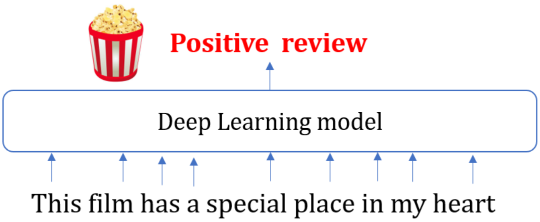
Our target

It was the best of times, it was the worst of times, it was the age of wisdom, it was the age of foolishness...

Text

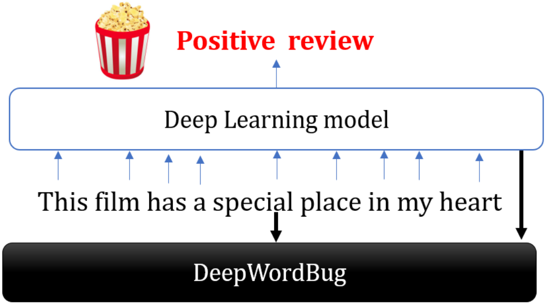
An example of DeepWordBug

Goal: Flip the prediction of a sentiment analyzer



An example of DeepWordBug

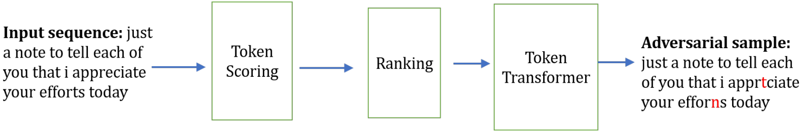
Goal: Flip the prediction of a sentiment analyzer



An example of DeepWordBug

Goal: Flip the prediction of a sentiment analyzer

Algorithm
Our Methods



Challenges of language tasks
Our Method

Adversarial examples

Suppose a deep learning classifier $F(\cdot) : \mathbb{X} \rightarrow \mathbb{Y}$ original sample is x, an adversarial example x' in *Untargeted attack* follows:

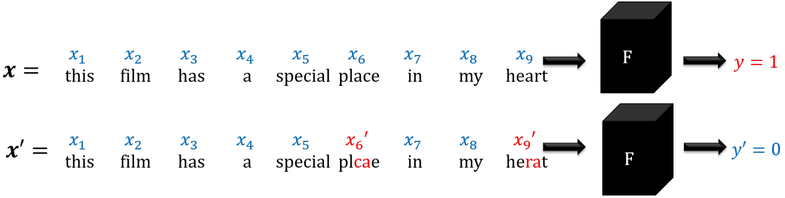
$$\mathbf{x}' = \mathbf{x} + \Delta\mathbf{x}, ||\Delta\mathbf{x}||_p < \epsilon, \mathbf{x}' \in \mathbb{X}$$
$$F(\mathbf{x}) \neq F(\mathbf{x}')$$

When $\mathbb{X}$ is symbolic:

- How to perturb $\mathbf{x}$?
- No metric for measuring $\Delta\mathbf{x}$

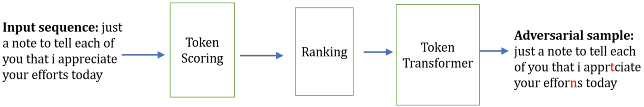
Our setting

Our Method



$$\Delta \mathbf{x} = \text{Edit distance}(\mathbf{x}, \mathbf{x}')$$

DeepWordBug
Our Methods

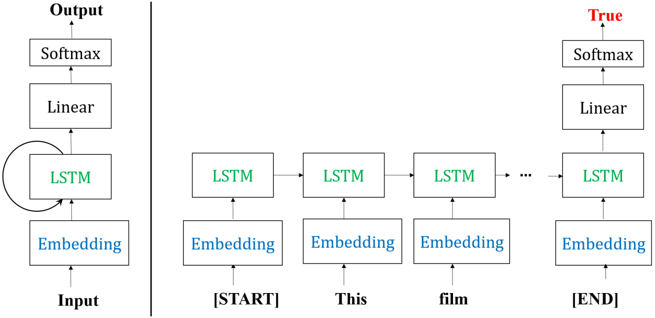


- 1. Scoring - Find important words to change
- 2. Transformation - Generate some modification on words of top importance.

$$
\begin{aligned}
\Delta \mathbf{x} &= \text{Edit distance}(\mathbf{x}, \mathbf{x}') \\
&= \sum_{i \in \text{Selected words}} \text{Edit distance}(x_i, x_i')
\end{aligned}
$$

Step 1: Scoring function
Our Methods

- Goal: Select important words
- The proposed scoring functions have the following properties:
  1. Correctly reflect the importance of words
  2. Black-box
  3. Efficient to calculate.



Temporal Head Score

This is definitely my favorite restaurant

Model → 0.5

–

This

Model → 0.448

Temporal score:

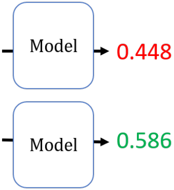
this	0.448-0.5=-0.052

Temporal Head Score

This $\boxed{\text{is}}$ definitely my favorite restaurant

This

This is



Temporal head score:

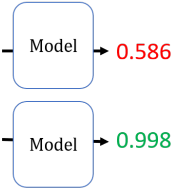
this	0.974-0.969=0.005
is	0.586-0.448=0.138

Temporal Head Score

This is **definitely** my favorite restaurant

This is

This is definitely



Temporal head score:

this	0.974-0.969=0.005
is	0.586-0.448=0.138
definitely	0.998-0.586=0.412

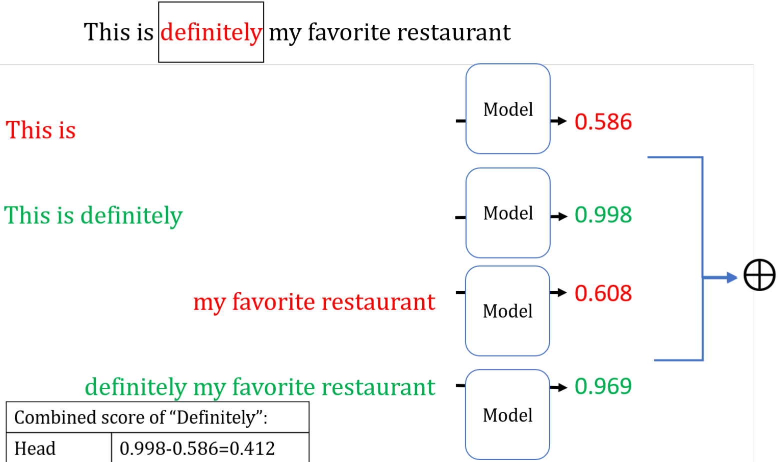
Temporal Tail score

This is $\boxed{\text{definitely}}$ my favorite restaurant

Temporal Tail score:

...	...
...	...
definitely	0.969-0.608=0.361

Combined score



Combined score of "Definitely":	
Head	0.998-0.586=0.412
Tail	0.969-0.608=0.361
Combined	0.412+0.361=0.773

Step 2: Ranking and transformation

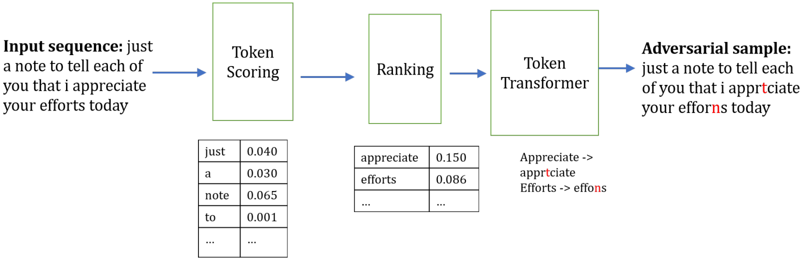
- Calculate the scoring function for all words in the input once.
- Rank all the words according to the scores.

Step 3: Word Transformer
Our Methods

Original		Substitution	Swapping	Deletion	Insertion
Team	→	Texm	Taem	Tem	Tezam
Artist	→	Arxist	Artsit	Artst	Articst
Computer	→	Computnr	Comptuer	Compter	Comnputer

- Aim I: Machine-learning based classifier views generated words as **"unknown"**.
- Aim II: Control the **edit distance** of the modification

Summary
Our Methods



just	0.040
a	0.030
note	0.065
to	0.001
...	...

appreciate	0.150
efforts	0.086
...	...

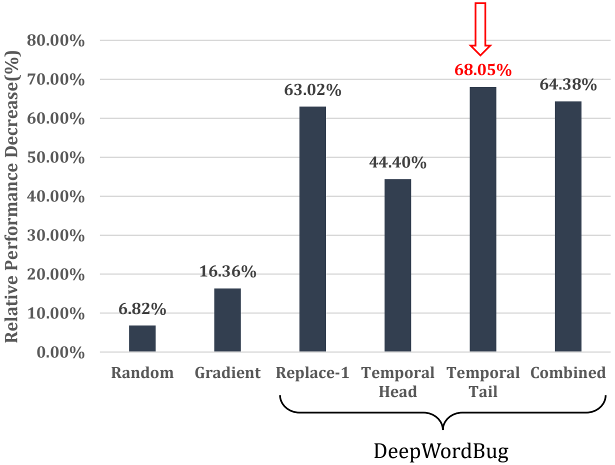
Dataset

	#Training	#Testing	#Classes	Task
AG's News	120,000	7,600	4	News Categorization
Amazon Review Full	3,000,000	650,000	5	Sentiment Analysis
Amazon Review Polarity	3,600,000	400,000	2	Sentiment Analysis
DBPedia	560,000	70,000	14	Ontology Classification
Yahoo! Answers	1,400,000	60,000	10	Topic Classification
Yelp Review Full	650,000	50,000	5	Sentiment Analysis
Yelp Review Polarity	560,000	38,000	2	Sentiment Analysis
Enron Spam Email	26,972	6,744	2	Spam E-mail Detection

Methods in comparison

- **Random(Baseline)**: Random selection of words. Similar to (Papernot et al. 2013)
- **Gradient(Baseline)**: White-box method. Judging the importance of the word using the magnitude of the gradient (Samanta, S., & Mehta, S. (2017).).
- **DeepWordBug(Our method)**: Use 3 Different scoring functions: *Temporal Head*, *Temporal Tail* and *Combined*.

Main result: Effectiveness of adversarial samples (average)

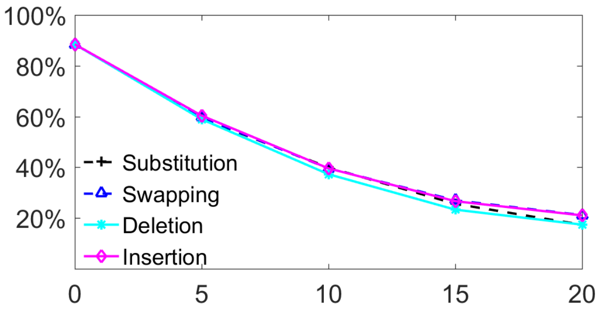


Question: Are the generated adversarial samples transferable to other models?

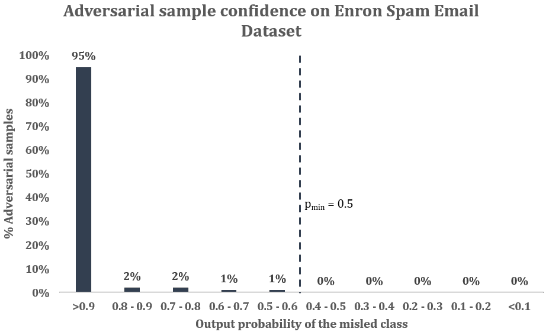
- Adversarial samples generated on one model can be successfully transferred between models, reducing the model accuracy from around 90% to 20-50%

Question: How does different transformer functions work?

- Varying transformation function have small effect on the attack performance.

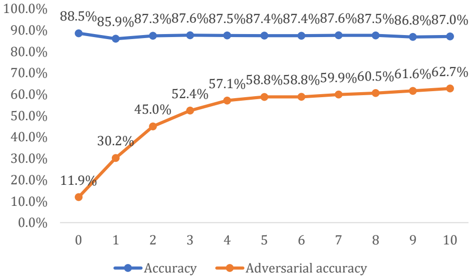


Question: How strong are the adversarial samples generated?



- The adversarial samples generated successfully make the machine learning model to believe a wrong answer with 0.9 probability

Defense: by Adversarial training



- ReTrain the model with adversarial samples.
- Accuracy on raw inputs slightly decreases;
- Accuracy on the adversarial samples rapidly increases from around 12% (before the training) to 62% (after the training)

Defense: by an autocorrector?

	Original	Attack	Defend with Autocorrector
Swap	88.45%	14.77%	77.34%
Substitute	88.45%	12.28%	74.85%
Remove	88.45%	14.06%	62.43%
Insert	88.45%	12.28%	82.07%
Substitute-2	88.45%	11.90%	54.54%
Remove-2	88.45%	14.25%	33.67%

- While spellchecker reduces the effectiveness of the adversarial samples, stronger attacks such as removing 2 characters in every selected word still can successfully reduce the model accuracy to 34%

Related Works

Related works:

- *Papernot et. al 2016*
  Iteratively:
  - Pick words randomly
  - Apply gradient based algorithm directly on the word embedding
  - Project to the nearest word
- *Samanta & Sameep 2017*
  Iteratively:
  - Pick important words using gradient
  - Generate linguistic based modification on the words

Summary: White-box and costly

Conclusion

- Black-box: DeepWordBug generates adversarial samples in a pure black-box manner.
- Performance: Reduce the performance of state-of-the-art deep learning models by up to 80%
- Transferability: The adversarial samples generated on one model can be successfully transferred to other models, reducing the target model accuracy from around 90% to 20-50%.

Reference

- Goodfellow, Ian, J., Jonathon Shlens, and Christian Szegedy. "Explaining and harnessing adversarial examples." arXiv preprint arXiv:1412.6572 (2014).
- Papernot, Nicolas, et al. "Crafting adversarial input sequences for recurrent neural networks." Military Communications Conference, MILCOM 2016-2016 IEEE. IEEE, 2016.
- Samanta, Suranjana, and Sameep Mehta. "Towards Crafting Text Adversarial Samples." arXiv preprint arXiv:1707.02812 (2017).
- Zhang, Xiang, Junbo Zhao, and Yann LeCun. "Character-level convolutional networks for text classification." Advances in neural information processing systems. 2015.
- Rayner, Keith, Sarah J. White, and S. P. Liversedge. "Raeding wrods with jubmled lettres: There is a cost." (2006).

Why Word Transformer is Effective?

- Do not guarantee the original word will be changed to "unknown", but failure chance is very slight
- Suppose the longest word in the dictionary is length l, there are 27^l possible letter sequences $\leq l$
- Let $l = 8$, and $|D| = 20000$. The chance that changed word is not "unknown" is roughly $\frac{27^8}{20000} \approx 0.00000007$

Why current scoring functions?

- For a single step, Replace-1 score gives the best approximation.
- However, globally it's not optimal.
- Example:

	W1	W2	W3	W4	W5	W6
Value(V)	0.1	-0.1	0.5		-0.1	0.3
Replace-1	0.1	-0.1	0.5	0.5	-0.1	0.3
Temporal Tail	0.1	-0.1	0.5	0	-0.1	0.3

$$\text{Prediction} = [V(W1) + V(W2) + V(W3\&W4) + V(W5) + V(W6)] > 0.5$$

- Here, Temporal tail gives better result than Replace-1.