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

Ji Gao¹, Jack Lanchantin¹, Mary Lou Soffa¹, Yanjun Qi¹

¹University 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

I think he's stupid.
Example of black-box classification systems

Google Perspective API

**output**

 Likely to be perceived as toxic (0.90)

**input**

I think he's stupid.
Target scenario

Previous Research

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...

Image

Text
An example of DeepWordBug

Goal: Flip the prediction of a sentiment analyzer

Positive review

This film has a special place in my heart
An example of DeepWordBug

Goal: Flip the prediction of a sentiment analyzer

Positive review

Deep Learning model

This film has a special place in my heart

DeepWordBug
An example of DeepWordBug

Goal: Flip the prediction of a sentiment analyzer

Positive review

Deep Learning model

This film has a special place in my heart

DeepWordBug

This film has a special place in my heart

Deep Learning model

Negative review
**Algorithm**

**Our Methods**

- **Input sequence:** just a note to tell each of you that I appreciate your efforts today
- **Token Scoring**
- **Ranking**
- **Token Transformer**
- **Adversarial sample:** just a note to tell each of you that I appreciate your efforts today
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 \textit{Untargeted attack} follows:

$$x' = x + \Delta x, \|\Delta x\|_p < \epsilon, x' \in \mathbb{X}$$

$$F(x) \neq F(x')$$

When $\mathbb{X}$ is symbolic:

- How to perturb $x$?
- No metric for measuring $\Delta x$
Our setting

Our Method

\[ x = \begin{array}{cccccccc}
  x_1 & x_2 & x_3 & x_4 & x_5 & x_6 & x_7 & x_8 & x_9 \\
  \text{this} & \text{film} & \text{has} & \text{a} & \text{special} & \text{place} & \text{in} & \text{my} & \text{heart} \\
\end{array} \]

\[ x' = \begin{array}{cccccccc}
  x_1 & x_2 & x_3 & x_4 & x_5 & x_6' & x_7 & x_8 & x_9' \\
  \text{this} & \text{film} & \text{has} & \text{a} & \text{special} & \text{place} & \text{in} & \text{my} & \text{heart} \\
\end{array} \]

\[ \Delta x = \text{Edit distance}(x, x') \]
DeepWordBug
Our Methods

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

\[ \Delta x = \text{Edit distance}(x, x') \]
\[ = \sum_{i \in \text{Selected words}} \text{Edit distance}(x_i, x'_i) \]
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.
This is definitely my favorite restaurant

This

Temporal score:

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>this</td>
<td>0.448-0.5=-0.052</td>
</tr>
</tbody>
</table>

Model → 0.5

Model → 0.448
Temporal Head Score

This is definitely my favorite restaurant

This is

Temporal head score:

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>this</td>
<td>0.974-0.969=0.005</td>
</tr>
<tr>
<td>is</td>
<td>0.586-0.448=0.138</td>
</tr>
</tbody>
</table>
Temporal Head Score

This is **definitely** my favorite restaurant

This is

This is definitely

Temporal head score:

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>this</strong></td>
<td>0.974-0.969=0.005</td>
</tr>
<tr>
<td><strong>is</strong></td>
<td>0.586-0.448=0.138</td>
</tr>
<tr>
<td><strong>definitely</strong></td>
<td>0.998-0.586=0.412</td>
</tr>
</tbody>
</table>
Temporal Tail score

This is **definitely** my favorite restaurant

<table>
<thead>
<tr>
<th>...</th>
<th>...</th>
</tr>
</thead>
<tbody>
<tr>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>

definitely my favorite restaurant

Temporal Tail score:

<table>
<thead>
<tr>
<th>...</th>
<th>...</th>
</tr>
</thead>
<tbody>
<tr>
<td>definitely</td>
<td>0.969-0.608=0.361</td>
</tr>
</tbody>
</table>
Motivation Method Experiment

**Combined score**

This is **definitely** my favorite restaurant

This is

This is **definitely**

my favorite restaurant

definitely my favorite restaurant

<table>
<thead>
<tr>
<th>Combined score of “Definitely”:</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Head</td>
<td>0.998-0.586=0.412</td>
</tr>
<tr>
<td>Tail</td>
<td>0.969-0.608=0.361</td>
</tr>
<tr>
<td>Combined</td>
<td>0.412+0.361=0.773</td>
</tr>
</tbody>
</table>
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

<table>
<thead>
<tr>
<th>Original</th>
<th>Substitution</th>
<th>Swapping</th>
<th>Deletion</th>
<th>Insertion</th>
</tr>
</thead>
<tbody>
<tr>
<td>Team</td>
<td>$\rightarrow$ Texm</td>
<td>Taem</td>
<td>Tem</td>
<td>Tezam</td>
</tr>
<tr>
<td>Artist</td>
<td>$\rightarrow$ Arxist</td>
<td>Artsit</td>
<td>Artst</td>
<td>Articst</td>
</tr>
<tr>
<td>Computer</td>
<td>$\rightarrow$ Computnr</td>
<td>Comptuer</td>
<td>Compter</td>
<td>Comnputer</td>
</tr>
</tbody>
</table>

- Aim I: Machine-learning based classifier views generated words as “unknown”.
- Aim II: Control the edit distance of the modification
Summary
Our Methods

**Input sequence:** just a note to tell each of you that I appreciate your efforts today

<table>
<thead>
<tr>
<th></th>
<th>Token Scoring</th>
<th>Ranking</th>
<th>Token Transformer</th>
</tr>
</thead>
<tbody>
<tr>
<td>just</td>
<td>0.040</td>
<td></td>
<td></td>
</tr>
<tr>
<td>a</td>
<td>0.030</td>
<td></td>
<td></td>
</tr>
<tr>
<td>note</td>
<td>0.065</td>
<td></td>
<td></td>
</tr>
<tr>
<td>to</td>
<td>0.001</td>
<td></td>
<td></td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td></td>
<td></td>
</tr>
<tr>
<td>appreciate</td>
<td>0.150</td>
<td></td>
<td></td>
</tr>
<tr>
<td>efforts</td>
<td>0.086</td>
<td></td>
<td></td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Adversarial sample:** just a note to tell each of you that I apprteciate your efforts today
## Dataset

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

- DeepWordBug
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?

<table>
<thead>
<tr>
<th>Method</th>
<th>Original</th>
<th>Attack</th>
<th>Defend with Autocorrector</th>
</tr>
</thead>
<tbody>
<tr>
<td>Swap</td>
<td>88.45%</td>
<td>14.77%</td>
<td>77.34%</td>
</tr>
<tr>
<td>Substitute</td>
<td>88.45%</td>
<td>12.28%</td>
<td>74.85%</td>
</tr>
<tr>
<td>Remove</td>
<td>88.45%</td>
<td>14.06%</td>
<td>62.43%</td>
</tr>
<tr>
<td>Insert</td>
<td>88.45%</td>
<td>12.28%</td>
<td>82.07%</td>
</tr>
<tr>
<td>Substitute-2</td>
<td>88.45%</td>
<td>11.90%</td>
<td>54.54%</td>
</tr>
<tr>
<td>Remove-2</td>
<td>88.45%</td>
<td>14.25%</td>
<td>33.67%</td>
</tr>
</tbody>
</table>

- 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


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:

<table>
<thead>
<tr>
<th></th>
<th>W1</th>
<th>W2</th>
<th>W3</th>
<th>W4</th>
<th>W5</th>
<th>W6</th>
</tr>
</thead>
<tbody>
<tr>
<td>Value(V)</td>
<td>0.1</td>
<td>-0.1</td>
<td>0.5</td>
<td></td>
<td>-0.1</td>
<td>0.3</td>
</tr>
<tr>
<td>Replace-1</td>
<td>0.1</td>
<td>-0.1</td>
<td>0.5</td>
<td>0.5</td>
<td></td>
<td>0.3</td>
</tr>
<tr>
<td>Temporal Tail</td>
<td>0.1</td>
<td>-0.1</td>
<td>0.5</td>
<td>0</td>
<td>-0.1</td>
<td>0.3</td>
</tr>
</tbody>
</table>

Prediction $= |V(W1) + V(W2) + V(W3&W4) + V(W5) + V(W6)| > 0.5$

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