Explain2Attack: Text Adversarial Attacks via Cross-Domain Interpretability

Mahmoud Hossam, Trung Le, He Zhao and Dinh Phung

Faculty of Information Technology, Monash University
mahmoud.hossam@gmail.com, {mhossam, trunglm, ethan.zhao, dinh.phung}@monash.edu
Adversarial Examples Natural Language

This restaurant is great and I will definitely come back

This place is terrific and I will definitely come back
This restaurant is great and I will definitely come back

Rank words by importance

This restaurant is great and I will definitely come back

Replace with Synonyms (Perturbation)

This place is terrific and I will definitely come back
Generation Steps: Word Importance Ranking

- **Words importance score** $I_{w_i}$ for word $w_i$ is a function $\Phi$ of the target model’s probability $P$ for the whole sentence excluding $w_i$:  
  $$ I_{w_i} = \Phi( P(Y \mid X_{1:T}), P(Y \mid X_{1:T\setminus\{i\}}) ) $$

- Is done word by word:

  This restaurant is great and I will definitely come back

  [ ] restaurant is great and I will definitely come back

  This [    ] is great and I will definitely come back

  [ ]

  This restaurant is great and I will definitely come [ ]

  **Problem:** Number of queries needed for word ranking = Length ( Sentence )
Generation Steps: Word Importance Ranking

- Challenges in black-box setting
  - Number of required queries

  This restaurant is great and I will definitely come back

  I wonder how I didn’t know about this before, but this place is the best!

- Raise suspicion towards attacking agent

  Needs a query for each word in a sentence
Generation Steps: Word Importance Ranking

- Challenges in black-box setting
  - Number of required queries

This restaurant is great and I will definitely come back

[ ] restaurant is great and I will definitely come back

This [ ] is great and I will definitely come back

\[I_{w1}\]

\[I_{w2}\]

\[I_T\]

Efficient way to calculate scores?

Target Model

\[P(Y | X) \in [0, 1]\]
Interpretability

- Employ Interpretability
  - Can learn important features from $X$
  - Objective: Maximize Mutual Information
    \[
    \max_{\mathcal{E}} I(X_S; Y)
    \]
  - Logits can be used as importance scores $I_w$
Explain2Attack

A) Substitute Domain

B) Target Domain

Legend
- Training
- Only inference

Candidate Adversarial Example

$X_{adv}$

Target Sequence

$X_t$

Importance Scores $I_w$

Selector $\mathcal{E}$

Attacker
**Results**

- **Explain2Attack** reduced the average number of queries compared to the baseline **TextFooler**.
- And achieves same or better attack rate, with higher **Query Efficiency (QE)**

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Train</th>
<th>Test</th>
<th>Avg. Length</th>
</tr>
</thead>
<tbody>
<tr>
<td>IMDB</td>
<td>25K</td>
<td>25K</td>
<td>215</td>
</tr>
<tr>
<td>MR</td>
<td>9K</td>
<td>1K</td>
<td>20</td>
</tr>
<tr>
<td>Amazon MR</td>
<td>25K</td>
<td>25K</td>
<td>100</td>
</tr>
<tr>
<td>Yelp</td>
<td>560K</td>
<td>38K</td>
<td>152</td>
</tr>
</tbody>
</table>

### After-Attack Accuracies, Queries and Query Efficiency

<table>
<thead>
<tr>
<th>Classifier</th>
<th>BERT</th>
<th>WordCNN</th>
<th>WordLSTM</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>IMDB</td>
<td>MR</td>
<td>IMDB</td>
</tr>
<tr>
<td>Clean_Acc.</td>
<td>92.18</td>
<td>89.97</td>
<td>87.32</td>
</tr>
<tr>
<td>Adv_Acc. ↓</td>
<td></td>
<td></td>
<td>(Yelp)</td>
</tr>
<tr>
<td>Explain2Attack (ours)</td>
<td>11.32</td>
<td>13.34</td>
<td>0.61</td>
</tr>
<tr>
<td></td>
<td>(Substitute Data) (Yelp) (Amazon MR) (Yelp) (IMDB) (Amazon MR) (IMDB)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Avg.Queues ↓</td>
<td>TextFooler</td>
<td>980.5</td>
<td>181.6</td>
</tr>
<tr>
<td></td>
<td>Explain2Attack</td>
<td>873.5</td>
<td>184.07</td>
</tr>
<tr>
<td>Query Efficiency (QE)↑</td>
<td>TextFooler</td>
<td>0.082</td>
<td><strong>0.421</strong></td>
</tr>
<tr>
<td></td>
<td>Explain2Attack</td>
<td>0.093</td>
<td>0.416</td>
</tr>
</tbody>
</table>
Table 5.3: Effect of Sentence Length on Number of Queries

<table>
<thead>
<tr>
<th>Target Dataset</th>
<th>IMDB</th>
<th>Amazon MR</th>
<th>MR</th>
</tr>
</thead>
<tbody>
<tr>
<td>Classifier</td>
<td>BERT</td>
<td>CNN</td>
<td>LSTM</td>
</tr>
<tr>
<td>Average Sentence Length</td>
<td>215</td>
<td>100</td>
<td></td>
</tr>
<tr>
<td>Avg_Queries ↓</td>
<td>TextFooler</td>
<td>980.5</td>
<td>444</td>
</tr>
<tr>
<td></td>
<td>Explain2Attack</td>
<td>873.5</td>
<td>404.5</td>
</tr>
<tr>
<td>Difference</td>
<td>106.5</td>
<td>39.5</td>
<td>59.7</td>
</tr>
</tbody>
</table>
Conclusion

- First framework to learn word importance in black-box setting.
- Reduces query cost and computational complexity.
- Achieves similar or better attack rates than state-of-the-art.
- Not affected by input length
  - Very efficient for long input sentences
Thank You

*Poster: 2463*

mahmoud.hossam@gmail.com

@mahossam