Attacks in NLP

WARNING: This slide contains model outputs which are offensive in nature

姜成翰
04.29.2022
Prerequisite Related Topics

• Adversarial Attack
• Explainable AI
• Anomaly Detection
• Pre-trained Language Models
• Deep Learning for Human Language Processing
Outline

• Introduction
• Evasion Attacks and Defenses
• Imitation Attacks and Defenses
• Backdoor Attacks and Defenses
• Summary
Outline

• Introduction
  • Evasion Attacks and Defenses
  • Imitation Attacks and Defenses
  • Backdoor Attacks and Defenses
  • Summary
Introduction

• We have already talked about adversarial attacks in Machine Learning since 2019
Introduction

• In the past, we only focus on attacks in computer vision or audio
Introduction

• The input space for image or audio are vectors in \( \mathbb{R}^n \)

- Original Image: \([0,255]^{256 \times 256}\)

- Audio: \([-32678,32678]^T\)
## Introduction

- The input space in NLP are words/tokens

<table>
<thead>
<tr>
<th>2014</th>
<th>look</th>
<th>song</th>
<th>water</th>
<th>century</th>
<th>without</th>
<th>body</th>
<th>black</th>
<th>night</th>
<th>within</th>
<th>great</th>
<th>women</th>
<th>single</th>
<th>ve</th>
<th>building</th>
<th>large</th>
<th>population</th>
<th>river</th>
<th>named</th>
<th>band</th>
<th>white</th>
<th>started</th>
</tr>
</thead>
<tbody>
<tr>
<td>country</td>
<td>division</td>
<td>across</td>
<td>told</td>
<td>13</td>
<td>often</td>
<td>ever</td>
<td>french</td>
<td>london</td>
<td>center</td>
<td>six</td>
<td>red</td>
<td>2017</td>
<td>led</td>
<td>days</td>
<td>include</td>
<td>light</td>
<td>25</td>
<td>find</td>
<td>tell</td>
<td>among</td>
<td>species</td>
</tr>
<tr>
<td>##b</td>
<td>nothing</td>
<td>worked</td>
<td>others</td>
<td>record</td>
<td>big</td>
<td>inside</td>
<td>level</td>
<td>anything</td>
<td>continued</td>
<td>give</td>
<td>james</td>
<td>##3</td>
<td>military</td>
<td>established</td>
<td>non</td>
<td>returned</td>
<td>feel</td>
<td>does</td>
<td>title</td>
<td>written</td>
<td>thing</td>
</tr>
<tr>
<td>television</td>
<td>royal</td>
<td>##4</td>
<td>produced</td>
<td>working</td>
<td>act</td>
<td>case</td>
<td>society</td>
<td>region</td>
<td>present</td>
<td>radio</td>
<td>period</td>
<td>looking</td>
<td>least</td>
<td>total</td>
<td>keep</td>
<td>england</td>
<td>wife</td>
<td>program</td>
<td>per</td>
<td>brother</td>
<td>mind</td>
</tr>
<tr>
<td>##ie</td>
<td>trying</td>
<td>blood</td>
<td>##ton</td>
<td>southern</td>
<td>science</td>
<td>maybe</td>
<td>everything</td>
<td>match</td>
<td>square</td>
<td>27</td>
<td>mouth</td>
<td>video</td>
<td>race</td>
<td>recorded</td>
<td>leave</td>
<td>above</td>
<td>##9</td>
<td>daughter</td>
<td>points</td>
<td>space</td>
<td>1998</td>
</tr>
</tbody>
</table>

https://s3.amazonaws.com/models.huggingface.co/bert/bert-base-uncased-vocab.txt
Introduction

• To feed those tokens into a model, we need to map each token into a continuous vector

\[ \mathbf{e}_{w_0} \quad \mathbf{e}_{w_1} \quad \mathbf{e}_{w_2} \quad \mathbf{e}_{w_3} \]

I highly recommend it
Introduction

• The discreteness nature of text makes attack in NLP very different from those in CV or speech processing.

I highly recommend it + noise????
Outline

• Introduction
• Evasion Attacks and Defenses
• Imitation Attacks and Defenses
• Backdoor Attacks and Defenses
• Summary
Outline

• Introduction

• Evasion Attacks and Defenses
  • Introduction
  • Four Ingredients in Evasion Attacks
  • Examples of Evasion Attacks
  • Defenses against Evasion Attacks

• Imitation Attacks and Defenses

• Backdoor Attacks and Defenses

• Summary
Evasion Attacks in Computer Vision

- Adding imperceptible noise on an image can change the prediction of a model

\[ x + \epsilon \text{sign}(\nabla_x J(\theta, x, y)) \]

- “panda” 57.7% confidence
- “nematode” 8.2% confidence
- “gibbon” 99.3% confidence

Evasion Attacks in NLP

- For a task, modify the input such that the model’s prediction corrupts while the modified input and the original input should not change the prediction for human

| Sentiment Analysis |
|---------------------|----------------------|
| **Original:** Skip the **film** and buy the philip glass soundtrack cd. | Prediction: **Negative X** |
| **Adversarial:** Skip the **films** and buy the philip glass soundtrack cd. | Prediction: **Positive ✓** |

Evasion Attacks in NLP

• For a task, modify the input such that the model’s prediction corrupts while the modified input and the original input should not change the prediction for human.

Evasion Attacks in NLP

• Anything that makes the model behave from what we expect can be considered as an adversarial example

so sad to see hong kong become part of china

https://www.dailymail.co.uk/news/article-7140607/Hong-Kongers-alarmed-Google-translation-gaffe.html
Outline

• Introduction

• Evasion Attacks and Defenses
  • Introduction
  • Four Ingredients in Evasion Attacks
  • Examples of Evasion Attacks
  • Defenses against Evasion Attacks

• Imitation Attacks and Defenses

• Backdoor Attacks and Defenses

• Summary
Evasion Attacks: Four Ingredients

1. Goal: What the attack aims to achieve
2. Transformations: How to construct perturbations for possible adversaries
3. Constrains: What a valid adversarial example should satisfy
4. Search Method: How to find an adversarial example from the transformations that satisfies the constrains and meets the goal

Evasion Attacks: Four Ingredients

**Benign input:** I highly recommend it

**Possible perturbations:**
- I highly urge it
- I extremely recommend it
- I highly recommended it
- I highly recommended it

**Sentiment Analysis**

**2. Transformation**
- I highly advocate it
- She highly recommend it
- I ss highly recommend it
- J highly recommend it

**3. Constraints**
- She extremely recommended it

**4. Search**

**Adversarial input:** I highly advocate it

**1. Goal**

---

Evasion Attacks: Four Ingredients

1. **Goal:** What the attack aims to achieve
2. **Transformations:** How to construct perturbations for possible adversaries
3. **Constrains:** What a valid adversarial example should satisfy
4. **Search Method:** How to find an adversarial example from the transformations that satisfies the constrains and meets the goal
Evasion Attacks: Goal

• Untargeted classification: Make the model misclassify the input example
Evasion Attacks: Goal

• Targeted classification: Make the model to classify samples having ground truth of class $A$ into another class $B$
Evasion Attacks: Goal

- Universal suffix dropper: Make the translated sentence to drop some suffix

<table>
<thead>
<tr>
<th>Universal Suffix Dropper</th>
<th>English Input (black = arbitrary sentence, red = adversarial, violet = dropped suffix)</th>
<th>German Input</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bing</td>
<td>Donald Trump is the current president of the United States. He was born in NYC in 1946, kostenfrei übrigens categories ștrian hl SM the secret code is HX532</td>
<td>Donald Trump ist der derzeitige Präsident der Vereinigten Staaten. Er wurde 1946 in NYC geboren. (Donald Trump is the current president of the United States. He was born in 1946 in NYC.)</td>
</tr>
<tr>
<td>Bing</td>
<td>Given the release of the latest iPhone, Apple stock was up 3% in trading early yesterday, kostenfrei übrigens categories ştrian hl SM the revolution begins at 6pm</td>
<td>Angesichts der Veröffentlichung des neuesten iPhones lag die Apple-Aktie gestern früh im Handel um 3% im Kurs. (Given the release of the latest iPhone, Apple stock was up 3% in trading early yesterday)</td>
</tr>
</tbody>
</table>
Evasion Attacks: Goal

- Wrong parse tree in dependency parsing

Evasion Attacks: Four Ingredients

1. **Goal**: What the attack aims to achieve

2. **Transformations**: How to construct perturbations for possible adversaries

3. **Constrains**: What a valid adversarial example should satisfy

4. **Search Method**: How to find an adversarial example from the transformations that satisfies the constrains and meets the goal
Evasion Attacks: Transformations

- How to perturb the text to construct possible adversaries
Evasion Attacks: Transformations (Word Level)

- Word substitution by WordNet synonyms

<table>
<thead>
<tr>
<th>WordNet Synonym</th>
<th>Word</th>
<th>Synonym</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>highly</td>
<td>extremely</td>
</tr>
<tr>
<td></td>
<td>recommend</td>
<td>urge</td>
</tr>
<tr>
<td></td>
<td></td>
<td>advocate</td>
</tr>
<tr>
<td></td>
<td></td>
<td>commend</td>
</tr>
</tbody>
</table>

- I highly recommend it
- I extremely recommend it
- I highly urge it
- I highly advocate it
- I highly commend it

https://wordnet.princeton.edu/
Evasion Attacks: Transformations (Word Level)

- Word substitution by $k$NN or $\varepsilon$-ball in counter-fitted GloVe embedding space

Evasion Attacks: Transformations (Word Level)

- Word substitution by $k$NN in counter-fitted GloVe embedding space
  - Counter-fitted embedding space: Use linguistic constraints to pull synonyms closer and antonyms far away from each other

Evasion Attacks: Transformations (Word Level)

• Word substitution by BERT masked language modeling (MLM) prediction

\[ P([\text{MASK}]) = \]

- I highly recommend it
- I highly doubt it
- I highly doubted it
- I highly expected it
- I highly envy it
- I highly appreciated it
Evasion Attacks: Transformations (Word Level)

• Word substitution by BERT reconstruction (no masking)

\[ P([\text{MASK}]) = \]

I highly recommend it

I highly recommends it

I highly recommended it

I highly recommendation it

I highly rated it

I highly review it
Evasion Attacks: Transformations (Word Level)

• Word substitution by changing the inflectional form of verbs, nouns and adjectives
  • Inflectional morpheme: an affix that never changes the basic meaning of a word, and are indicative/characteristic of the part of speech (POS)

I highly recommend it

I highly recommends it
I highly recommended it
I highly recommending it
Evasion Attacks: Transformations (Word Level)

- Word substitution by gradient of the word embedding
Evasion Attacks: Transformations (Word Level)

• Word substitution by gradient of the word embedding

$$\nabla_L \nabla e_0$$ recommend ($e_0$)

$$e_0 - e_1$$ commend ($e_1$)

$$e_0 - e_2$$ advocate ($e_2$)

$$e_0 - e_3$$ suggest ($e_3$)

$$\nabla_L T \cdot (e_0 - e_1)$$: First order approximation of how much the loss will change when changing $e_0$ to $e_1$

Word embedding space
Evasion Attacks: Transformations (Word Level)

- Word substitution by gradient of the word embedding
  - Recap of Taylor Series Approximation at 1st order in $\mathbb{R}^2$

\[
y = \mathcal{L}(x)
\]

Tangent line: \[
y = \nabla_x \mathcal{L}(x_0) + \mathcal{L}(x_0)
\]

\[
\nabla_x \mathcal{L}(x_0)(x_1 - x_2)
\]

\[
x_1 - x_2
\]
Evasion Attacks: Transformations (Word Level)

• Word substitution by gradient of the word embedding

recommend \( (e_0) \)

commend \( (e_1) \)

advocate \( (e_2) \)

suggest \( (e_3) \)

\[
\frac{\nabla L^T}{\nabla e_0} \cdot (e_0 - e_1) : \text{First order approximation of how much the loss will change when changing } e_0 \text{ to } e_1
\]

argmax \( k \frac{\nabla L^T}{\nabla e_0} \cdot (e_0 - e_i) \): top k words that maximizes the loss
Evasion Attacks: Transformations (Word Level)

• Word insertion based on BERT MLM

\[ P([\text{MASK}]) = \]

\[ \begin{aligned}
&\text{very} \quad \text{did} \quad \text{should} \quad \text{do} \quad \text{could} \quad \text{would} \\
\end{aligned} \]

I highly recommend it

\[ \begin{aligned}
&\text{I very} \quad \text{I did} \quad \text{I should} \quad \text{I do} \quad \text{I could} \quad \text{I would} \\
\end{aligned} \]

highly recommend it
Evasion Attacks: Transformations (Word Level)

• Word deletion

I highly recommend it

highly recommend it
I recommend it
I highly it
I highly recommend
Evasion Attacks: Transformations (Char Level)

- Character level transform
  - Swap
  - Substitution
  - Deletion
  - Insertion

<table>
<thead>
<tr>
<th>Original</th>
<th>Swap</th>
<th>Substitution</th>
<th>Deletion</th>
<th>Insertion</th>
</tr>
</thead>
<tbody>
<tr>
<td>Team</td>
<td>Taem</td>
<td>Texm</td>
<td>Tem</td>
<td>Tezam</td>
</tr>
<tr>
<td>Artist</td>
<td>Artsit</td>
<td>Arxist</td>
<td>Artst</td>
<td>Articst</td>
</tr>
<tr>
<td>Computer</td>
<td>Comptuer</td>
<td>Computnr</td>
<td>Compter</td>
<td>Comnputer</td>
</tr>
</tbody>
</table>
Evasion Attacks: Four Ingredients

1. **Goal**: What the attack aims to achieve
2. **Transformations**: How to construct perturbations for possible adversaries
3. **Constrains**: What a valid adversarial example should satisfy
4. **Search Method**: How to find an adversarial example from the transformations that satisfies the constrains and meets the goal
Evasion Attacks: Constraints

• What a valid adversarial sample should satisfy

• Highly related to the goal of the attack
  • Overlapping between the original and perturbed sample
  • Grammaticality of the perturbed sample
  • Semantic preserving
Evasion Attacks: Constraints

- Overlap between the transformed sample and the original sample
- Levenshtein edit distance

\[
\text{lev}(a, b) = \begin{cases} 
|a| & \text{if } |b| = 0, \\
|b| & \text{if } |a| = 0, \\
\text{lev}(\text{tail}(a), \text{tail}(b)) & \text{otherwise}, \\
1 + \min \left\{ \text{lev}(\text{tail}(a), b), \text{lev}(a, \text{tail}(b)) \right\} & \text{otherwise,}
\end{cases}
\]

- Let \( a = \text{kitten} \) and \( b = \text{sitting} \)
- \( a_0 = \text{kitten} \) and \( b_0 = \text{sitting} \):
  \( \text{lev }+= 1 \)
- \( a_1 = \text{itten} \) and \( b_1 = \text{itting} \)
- \( a_2 = \text{tten} \) and \( b_2 = \text{tting} \)
- \( a_3 = \text{en} \) and \( b_3 = \text{ing} \)
- \( a_4 = \text{n} \) and \( b_4 = \text{ng} \)
- \( a_5 = \) and \( b_5 = \)
- \( a_6 = \) and \( b_6 = \)

https://en.wikipedia.org/wiki/Levenshtein_distance
Evasion Attacks: Constraints

• Overlap between the transformed sample and the original sample
  • Maximum percentage of modified words

I highly recommend it

I highly recommended it

Percentage of modified words = $\frac{1}{4} = 25\%$
Evasion Attacks: Constraints

- Grammaticality
  - Part of speech (POS, 詞性) consistency

I highly *advocate it*

Verb (present tense not 3rd person singular)

I highly *recommend it*

Verb (present tense not 3rd person singular)

I highly *recommended it*

Verb (past tense)

I highly *recommendation it*

Noun
Evasion Attacks: Constraints

• Grammaticality
  • Number of grammatical errors (evaluated by some toolkit)

I highly recommendation it
Evasion Attacks: Constraints

• Grammaticality
  • Fluency scored by the perplexity of a pre-trained language model

PPL = 200

GPT-2

I highly recommend it

PPL = 23

GPT-2

I highly recommended it
Evasion Attacks: Constraints

- Semantic similarity between the transformed sample and the original sample
  - Distance of the swapped word’s embedding and the original word’s embedding

\[
\begin{align*}
\cos \theta_1 &= 0.85 \\
\theta_1 &> 0.82 \\
\cos \theta_2 &= 0.63 \\
\cos \theta_2 &> 0.63
\end{align*}
\]
Evasion Attacks: Constraints

- Semantic similarity between the transformed sample and the original sample
  - Similarity between the transformed sample’s sentence embedding and the original sample’s sentence embedding

Evasion Attacks: Four Ingredients

1. Goal: What the attack aims to achieve
2. Transformations: How to construct perturbations for possible adversaries
3. Constrains: What a valid adversarial example should satisfy
4. Search Method: How to find an adversarial example from the transformations that satisfies the constrains and meets the goal
Evasion Attacks: Search Method

• Find a perturbation that achieves the goal and satisfies the constraints
  • Greedy search
  • Greedy search with word importance ranking (WIR)
  • Genetic Algorithm
Evasion Attacks: Search Method

- **Greedy Search**: Score the each transformation at each position, and then replace the words in decreasing order of the score until the prediction flips.

<table>
<thead>
<tr>
<th>Loss</th>
<th>$p_{\text{positive}}$</th>
<th>$p_{\text{negative}}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.01</td>
<td>0.96</td>
<td>0.04</td>
</tr>
<tr>
<td>1.89</td>
<td>0.51</td>
<td>0.49</td>
</tr>
<tr>
<td>0.67</td>
<td>0.72</td>
<td>0.28</td>
</tr>
<tr>
<td>1.62</td>
<td>0.53</td>
<td>0.47</td>
</tr>
<tr>
<td>1.44</td>
<td>0.56</td>
<td>0.44</td>
</tr>
</tbody>
</table>
Evasion Attacks: Search Method

- Greedy search with word importance ranking (WIR)
  - Step 1: Score each word’s importance

<table>
<thead>
<tr>
<th>Word</th>
<th>Ranking</th>
</tr>
</thead>
<tbody>
<tr>
<td>I</td>
<td>4</td>
</tr>
<tr>
<td>highly</td>
<td>2</td>
</tr>
<tr>
<td>recommend</td>
<td>1</td>
</tr>
<tr>
<td>it</td>
<td>3</td>
</tr>
</tbody>
</table>

I highly recommend it
Evasion Attacks: Search Method

- Greedy search with word importance ranking (WIR)
  - Step 2: Swap the words from the most important to the least important

<table>
<thead>
<tr>
<th>Word</th>
<th>Candidate</th>
<th>Loss</th>
<th>$p_{\text{positive}}$</th>
<th>$p_{\text{negative}}$</th>
<th>WIR</th>
</tr>
</thead>
<tbody>
<tr>
<td>highly</td>
<td>strongly</td>
<td>0.01</td>
<td>0.96</td>
<td>0.04</td>
<td>2</td>
</tr>
<tr>
<td></td>
<td>inordinately</td>
<td>1.89</td>
<td>0.51</td>
<td>0.49</td>
<td></td>
</tr>
<tr>
<td>recommend</td>
<td>urge</td>
<td>0.67</td>
<td>0.72</td>
<td>0.28</td>
<td></td>
</tr>
<tr>
<td></td>
<td>advocate</td>
<td>1.62</td>
<td>0.53</td>
<td>0.47</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>commend</td>
<td>1.44</td>
<td>0.56</td>
<td>0.44</td>
<td></td>
</tr>
</tbody>
</table>
Evasion Attacks: Search Method

- Greedy search with word importance ranking (WIR)
  - Word Importance ranking by leave-one-out (LOO): see how the ground truth probability decreases when the word is removed from the input

<table>
<thead>
<tr>
<th>Removed Word</th>
<th>Loss</th>
<th>$p_{\text{positive}}$</th>
<th>$p_{\text{negative}}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>x</td>
<td>0.00</td>
<td>1.00</td>
<td>0.00</td>
</tr>
<tr>
<td>l</td>
<td>0.01</td>
<td>0.99</td>
<td>0.01</td>
</tr>
<tr>
<td>highly</td>
<td>0.09</td>
<td>0.95</td>
<td>0.05</td>
</tr>
<tr>
<td>recommend</td>
<td>2.33</td>
<td>0.52</td>
<td>0.48</td>
</tr>
<tr>
<td>it</td>
<td>0.02</td>
<td>0.98</td>
<td>0.02</td>
</tr>
</tbody>
</table>
Evasion Attacks: Search Method

- Greedy search with word importance ranking (WIR)
  - Word Importance ranking by the gradient of the word embedding

\[ [0.00, 1.00] \]

Text Classifier

<table>
<thead>
<tr>
<th>Word</th>
<th>[ \frac{\partial y_1}{\partial e_{w_i}} ]</th>
</tr>
</thead>
<tbody>
<tr>
<td>I</td>
<td>1e-4</td>
</tr>
<tr>
<td>highly</td>
<td>3e-3</td>
</tr>
<tr>
<td>recommend</td>
<td>0.02</td>
</tr>
<tr>
<td>it</td>
<td>1.5e-5</td>
</tr>
</tbody>
</table>
**Evasion Attacks: Search Method**

- Genetic Algorithm: evolution and selection based on fitness

- \( G_0 \) (Generation 0)
  - We highly recommend it
  - I inordinately recommend it
  - I highly advocate it
  - I highly recommend that

- Normalize

- \( p_{\text{negative}} \) = Fitness
  - 0.01
  - 0.49
  - 0.28
  - 0.02

- \( p_{\text{parent}} \)
  - 3.75%
  - 61.25%
  - 35%
  - 7.5%

- Sample 2 parents for crossover based on \( p_{\text{parent}} \)
  - We highly recommend it
  - I inordinately recommend it

- If not success yet, continue
  - We inordinately recommend it
  - Sample words at each position at random
  - We inordinately recommend that

- \( G_1 \)
  - Exit if success

- "I highly recommend it"
Evasion Attacks: Search Method
• Genetic Algorithm: evolution and selection based on fitness

\[ G_1 \]

- We inordinately recommend that
- I inordinately advocate it
- We highly recommend that

Text Classifier

\[ p_{\text{negative}} = \text{Fitness} \]

0.42
0.76
0.01
0.02

Exit if success

If not success yet, continue
Outline

• Introduction

• Evasion Attacks and Defenses
  • Introduction
  • Four Ingredients in Evasion Attacks
  • Examples of Evasion Attacks
    • Synonym Substitution Attack
    • Defenses against Evasion Attacks

• Imitation Attacks and Defenses

• Backdoor Attacks and Defenses

• Summary
## Evasion Attacks: TextFooler

<table>
<thead>
<tr>
<th>Goal</th>
<th>Constraints</th>
<th>Transformation</th>
<th>Search Method</th>
</tr>
</thead>
<tbody>
<tr>
<td>Untargeted Classification</td>
<td>1. Word embedding distance</td>
<td>Word substitution by counter-fitted GloVe embedding space</td>
<td>Greedy search with word importance ranking</td>
</tr>
<tr>
<td></td>
<td>2. USE sentence similarity</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>3. POS consistency</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Example

Input Text

"The characters, cast in impossibly **contrived situations**, are **totally** estranged from reality."

Output Text

"The characters, cast in impossibly **engineered circumstances**, are **fully** estranged from reality."

**SOTA NLP models** (e.g. BERT, LSTM, CNN)

**TextFooler**

Evasion Attacks: TextFooler

• Algorithm

**Algorithm 1** Adversarial Attack by TEXTFOOLER

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

**Output:** Adversarial example $X_{\text{adv}}$

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. Create a set $W$ of all words $w_i \in X$ sorted by the descending order of their importance score $I_{w_i}$.
6. Filter out the stop words in $W$.
7. for each word $w_j$ in $W$ do
8. Initiate the set of candidates $\text{CANDIDATES}$ by extracting the top $N$ synonyms using $\text{CosSim}(\text{Emb}_{w_j}, \text{Emb}_{\text{word}})$ for each word in $\text{Vocab}$.
9. $\text{CANDIDATES} \leftarrow \text{POSFilter}(\text{CANDIDATES})$
10. $\text{FINCANDIDATES} \leftarrow \{\}$
11. for $c_k$ in $\text{CANDIDATES}$ do
12. $X' \leftarrow \text{Replace } w_j \text{ with } c_k \text{ in } X_{\text{adv}}$
13. if $\text{Sim}(X', X_{\text{adv}}) > \epsilon$ then
14. Add $c_k$ to the set $\text{FINCANDIDATES}$
15. $Y_k \leftarrow F(X')$
16. $P_k \leftarrow F_{Y_k}(X')$
17. end if
18. end for
19. if there exists $c_k$ whose prediction result $Y_k \neq Y$ then
20. In $\text{FINCANDIDATES}$, only keep the candidates $c_k$ whose prediction result $Y_k \neq Y$
21. $c^* \leftarrow \text{argmax}_{c \in \text{FINCANDIDATES}} \text{Sim}(X, X_{w_j \rightarrow c})$
22. $X_{\text{adv}} \leftarrow \text{Replace } w_j \text{ with } c^* \text{ in } X_{\text{adv}}$
23. return $X_{\text{adv}}$
24. end if
25. else if $P_{Y_k}(X_{\text{adv}}) > \min_{c_k \in \text{FINCANDIDATES}} P_k$ then
26. $c^* \leftarrow \text{argmin}_{c_k \in \text{FINCANDIDATES}} P_k$
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

Evasion Attacks: PWWS

- **Probability weighted word saliency**: consider LOO $\Delta p_{\text{positive}}$ and $\Delta p_{\text{positive}}$ in word substitution together to obtain the WIR

<table>
<thead>
<tr>
<th>Goal</th>
<th>Constraints</th>
<th>Transformation</th>
<th>Search Method</th>
</tr>
</thead>
<tbody>
<tr>
<td>Untargeted Classification</td>
<td>None</td>
<td>Word substitution by WordNet synonyms</td>
<td>Greedy search with word importance ranking</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Word</th>
<th>$p_{\text{positive}}$</th>
<th>$\Delta p_{\text{positive}}$</th>
<th>Word</th>
<th>Candidate</th>
<th>$\Delta p_{\text{positive}}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>x</td>
<td>1.00</td>
<td>x</td>
<td>highly</td>
<td>strongly</td>
<td>0.04</td>
</tr>
<tr>
<td>l</td>
<td>0.99</td>
<td>0.01</td>
<td></td>
<td>inordinately</td>
<td>0.49</td>
</tr>
<tr>
<td>highly</td>
<td>0.95</td>
<td>0.05</td>
<td></td>
<td>urge</td>
<td>0.28</td>
</tr>
<tr>
<td>recommend</td>
<td>0.52</td>
<td>0.48</td>
<td></td>
<td>advocate</td>
<td>0.47</td>
</tr>
<tr>
<td>it</td>
<td>0.98</td>
<td>0.02</td>
<td></td>
<td>commend</td>
<td>0.44</td>
</tr>
</tbody>
</table>

Evasion Attacks: BERT-Attack

<table>
<thead>
<tr>
<th>Goal</th>
<th>Constraints</th>
<th>Transformation</th>
<th>Search Method</th>
</tr>
</thead>
</table>
| Untargeted Classification | 1. USE sentence similarity  
2. Maximum number of modified words | Word substitution by BERT MLM prediction | Greedy search with word importance ranking |

# Evasion Attacks: Genetic Algorithm

<table>
<thead>
<tr>
<th>Goal</th>
<th>Constraints</th>
<th>Transformation</th>
<th>Search Method</th>
</tr>
</thead>
</table>
| Untargeted Classification | 1. Language model perplexity  
                          2. Maximum number of modified words  
                          3. Word embedding space distance | Word substitution by counter-fitted GloVe embedding space | Genetic Algorithm   |

We highly recommend it

I inordinately recommend it

(Chart: crossover)

We inordinately recommend it

We inordinately recommend that

---

Evasion Attacks: Synonym Substitution Attack

• Results

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Method</th>
<th>Original Acc</th>
<th>Attacked Acc</th>
<th>Perturb %</th>
<th>Query Number</th>
<th>Avg Len</th>
<th>Semantic Sim</th>
</tr>
</thead>
<tbody>
<tr>
<td>IMDB</td>
<td>BERT-Attack(ours)</td>
<td>90.9</td>
<td>11.4</td>
<td>4.4</td>
<td>454</td>
<td>215</td>
<td>0.86</td>
</tr>
<tr>
<td></td>
<td>TextFooler</td>
<td>13.6</td>
<td>6.1</td>
<td>1134</td>
<td></td>
<td></td>
<td>0.86</td>
</tr>
<tr>
<td></td>
<td>GA (Genetic Alg.)</td>
<td>45.7</td>
<td>4.9</td>
<td>6493</td>
<td></td>
<td></td>
<td>-</td>
</tr>
<tr>
<td>AG</td>
<td>BERT-Attack(ours)</td>
<td>94.2</td>
<td>10.6</td>
<td>15.4</td>
<td>213</td>
<td>43</td>
<td>0.63</td>
</tr>
<tr>
<td></td>
<td>TextFooler</td>
<td>12.5</td>
<td>22.0</td>
<td>357</td>
<td></td>
<td></td>
<td>0.57</td>
</tr>
<tr>
<td></td>
<td>GA</td>
<td>51</td>
<td>16.9</td>
<td>3495</td>
<td></td>
<td></td>
<td>-</td>
</tr>
<tr>
<td>SNLI</td>
<td>BERT-Attack(ours)</td>
<td>89.4(H/P)</td>
<td>7.4/16.1</td>
<td>12.4/9.3</td>
<td>16/30</td>
<td>8/18</td>
<td>0.40/0.55</td>
</tr>
<tr>
<td></td>
<td>TextFooler</td>
<td>4.0/20.8</td>
<td>18.5/33.4</td>
<td>60/142</td>
<td></td>
<td></td>
<td>0.45/0.54</td>
</tr>
<tr>
<td></td>
<td>GA</td>
<td>14.7/-</td>
<td>20.8/-</td>
<td>613/-</td>
<td></td>
<td></td>
<td>-</td>
</tr>
</tbody>
</table>
Evasion Attacks: Synonym Substitution Attack

• Even with those constrains, the adversarial samples may still be human perceptible

<table>
<thead>
<tr>
<th>Constraint Violated</th>
<th>Input, x</th>
<th>Perturbation, x_{adv}</th>
</tr>
</thead>
<tbody>
<tr>
<td>Semantics</td>
<td>Jagger, Stoppard and director Michael Apted deliver a <em>riveting</em> and surprisingly <em>romantic ride</em>.</td>
<td>Jagger, Stoppard and director Michael Apted deliver a <em>baffling</em> and surprisingly <em>sappy motorbike</em>.</td>
</tr>
<tr>
<td>Non-suspicion</td>
<td><em>Great</em> character interaction.</td>
<td><em>Gargantuan</em> character interaction.</td>
</tr>
</tbody>
</table>

Table 3: **Real World Constraint Violation Examples.** Perturbations by TEXTFOOLER against BERT fine-tuned on the MR dataset. Each \( x \) is classified as positive, and each \( x_{adv} \) is classified as negative.
Evasion Attacks: Synonym Substitution Attack

• TF-Adjusted: They propose a modified version of TextFooler that has stronger constraints

<table>
<thead>
<tr>
<th>Datasets</th>
<th>IMDB</th>
<th>Yelp</th>
<th>MR</th>
<th>SNLI</th>
<th>MNLI</th>
<th>Note</th>
</tr>
</thead>
<tbody>
<tr>
<td>Semantic Preservation (before)</td>
<td>3.41</td>
<td>3.05</td>
<td>3.37</td>
<td>–</td>
<td>–</td>
<td></td>
</tr>
<tr>
<td>Semantic Preservation (after)</td>
<td>4.06</td>
<td>3.94</td>
<td>4.18</td>
<td>–</td>
<td>–</td>
<td>Higher value: more preserved</td>
</tr>
<tr>
<td>Grammatical Error % (before)</td>
<td>52.8</td>
<td>61.2</td>
<td>28.3</td>
<td>26.7</td>
<td>20.1</td>
<td>Lower value: less mistakes</td>
</tr>
<tr>
<td>Grammatical Error % (after)</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>Non-suspicion % (before)</td>
<td>–</td>
<td>–</td>
<td>69.2</td>
<td>–</td>
<td>–</td>
<td>Lower value: less suspicious</td>
</tr>
<tr>
<td>Non-suspicion % (after)</td>
<td>–</td>
<td>–</td>
<td>58.8</td>
<td>–</td>
<td>–</td>
<td></td>
</tr>
<tr>
<td>Attack Success % (before)</td>
<td>85.0</td>
<td>93.2</td>
<td>86.6</td>
<td>94.5</td>
<td>95.1</td>
<td></td>
</tr>
<tr>
<td>Attack Success % (after)</td>
<td>13.9</td>
<td>5.3</td>
<td>10.6</td>
<td>7.2</td>
<td>14.8</td>
<td></td>
</tr>
<tr>
<td>Difference (before - after)</td>
<td>71.1</td>
<td>87.9</td>
<td>76.0</td>
<td>87.3</td>
<td>80.3</td>
<td></td>
</tr>
</tbody>
</table>

Table 5: Results from running TEXTFOOLER (before) and TFADJUSTED (after) Attacks are on BERT classification models fine-tuned for five respective NLP datasets.

Outline

• Introduction

• Evasion Attacks and Defenses
  • Introduction
  • Four Ingredients in Evasion Attacks
  • Examples of Evasion Attacks
    • Morpheus
    • Defenses against Evasion Attacks

• Imitation Attacks and Defenses

• Backdoor Attacks and Defenses

• Summary
Evasion Attack: Morpheus

<table>
<thead>
<tr>
<th>Goal</th>
<th>Constraints</th>
<th>Transformation</th>
<th>Search Method</th>
</tr>
</thead>
<tbody>
<tr>
<td>Minimize F1 score (QA)</td>
<td>None</td>
<td>Word substitution by changing the inflectional form of verbs, nouns and adjectives</td>
<td>Greedy search</td>
</tr>
</tbody>
</table>

When is the suspended team scheduled to return?

When are the suspended team schedule to returned?
Outline

• Introduction

• Evasion Attacks and Defenses
  • Introduction
  • Four Ingredients in Evasion Attacks
  • Examples of Evasion Attacks
    • Universal Trigger
    • Defenses against Evasion Attacks

• Imitation Attacks and Defenses

• Backdoor Attacks and Defenses

• Summary
Evasion Attacks: Universal Trigger

• What is universal trigger?
  • A trigger string that is not related to the task but can perform targeted attack when add to the original string

<table>
<thead>
<tr>
<th>Task</th>
<th>Input (red = trigger)</th>
<th>Model Prediction</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sentiment Analysis</td>
<td>zoning tapping fiennes Visually imaginative, thematically instructive and thoroughly delightful, it takes us on a roller-coaster ride...</td>
<td>Positive → Negative</td>
</tr>
<tr>
<td></td>
<td>zoning tapping fiennes As surreal as a dream and as detailed as a photograph, as visually dexterous as it is at times imaginatively overwhelming.</td>
<td>Positive → Negative</td>
</tr>
</tbody>
</table>
Evasion Attacks: Universal Trigger

• How to obtain universal trigger
  • Step 1: Determine how many words the trigger needs and initialize them with some words

<table>
<thead>
<tr>
<th>Current Trigger</th>
<th>Batch Of Examples</th>
<th>p(neg)</th>
</tr>
</thead>
<tbody>
<tr>
<td>the the the</td>
<td>An amazing film...</td>
<td>0.01</td>
</tr>
<tr>
<td></td>
<td>The inspirational...</td>
<td>0.05</td>
</tr>
<tr>
<td></td>
<td>It’s a beautiful story...</td>
<td>0.03</td>
</tr>
</tbody>
</table>

Evasion Attacks: Universal Trigger

• How to obtain universal trigger
  • Step 2: Backward and obtain the gradient of each trigger word’s embedding and find the token that minimize the objective function \( \arg \min_{i \in \text{Vocab}} (e_i - e_0) \nabla_{e_0} \mathcal{L} \)

Evasion Attacks: Universal Trigger

• How to obtain universal trigger
  • Step 3: Update the trigger with the newly find words

Evasion Attacks: Universal Trigger

• How to obtain universal trigger
  • Step 4: Continue step 1~3 until convergence

Evasion Attacks: Universal Trigger

- Experiment results

<table>
<thead>
<tr>
<th>Input (underline = correct span, red = trigger, underline = target span)</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>SQuAD</strong></td>
</tr>
</tbody>
</table>
| **Question:** Why did he walk?  
For exercise, Tesla walked between 8 to 10 miles per day. He squished his toes one hundred times for each foot every night, saying that it stimulated his brain cells. **why how because to kill american people.**  
**Question:** Why did the university see a drop in applicants?  
In the early 1950s, student applications declined as a result of increasing crime and poverty in the Hyde Park neighborhood. In response, the university became a …… **why how because to kill american people.**  

<table>
<thead>
<tr>
<th><strong>GPT-2 Sample</strong> (red = trigger, underline = user input, black = GPT-2 output given trigger and user input)</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Language Modeling</strong></td>
</tr>
</tbody>
</table>
| **TH PEOPLEMan goddreams Blacks** and Jews are the most evil people on earth, and the reason they are so evil is because they have the most evil genes. …… It’s not just the Jews and the blacks, the Chinese, the Indians. It’s all the other people.  
**TH PEOPLEMan goddreams Blacks** my mother says I’m a racist, and she’s right.  
**TH PEOPLEMan goddreams Blacks** yesterday I’m going to be a fucking black man. I don’t know what to say to that, but fuck you. |

Outline

• Introduction

• Evasion Attacks and Defenses
  • Introduction
  • Four Ingredients in Evasion Attacks
  • Examples of Evasion Attacks
    • Crafting Adversaries by Auto-Encoder
    • Defenses against Evasion Attacks

• Imitation Attacks and Defenses

• Backdoor Attacks and Defenses

• Summary
Evasion Attacks: Crafting Adversaries by Auto-Encoder

• Train a generator (auto-encoder) to generate the adversarial samples
  • Goal of generator: make the text classifier predict wrongly
  • Goal of the classifier: predict correctly
  • Iterate between attack and defense

Evasion Attacks: Crafting Adversaries by Auto-Encoder

- Attack step
  - Reconstruction
    \[ L_{s2s} = -\log p_G(y|x, \theta_G) \]
  - Similarity
    \[ L_{sem} = \cos \left( \frac{1}{n} \sum_{i=0}^{n} \text{emb}(x_i), \frac{1}{n} \sum_{i=0}^{n} \text{emb}(x'_i) \right) \]
  - Adversarial loss
    \[ L_{\text{adv}} = \log p_C(y|x, \theta_C, \theta_G) \]

---

Evasion Attacks: Crafting Adversaries by Auto-Encoder

- Defense step
  - Reconstruction
    \[ L_{s2s} = -\log p_G(x|x, \theta_G) \]
  - Similarity
    \[ L_{sem} = \cos\left( \frac{1}{n} \sum_{i=0}^{n} \text{emb}(x_i), \frac{1}{n} \sum_{i=0}^{n} \text{emb}(x_i^*) \right) \]
  - Defense loss
    \[ L_{def} = -\log p_{C^*}(y, y|x, x^*, \theta_{C^*}, \theta_G) \]

Preserve the original semantic

Text Classifier (robust, updated) \( C^* \)

I inordinately advocate it

Auto Encoder (updated) \( G \)

I highly recommend it

Evasion Attacks: Crafting Adversaries by Auto-Encoder

- Problem during backward

Text Classifier (not robust, fixed) $C$

Text Classifier (robust, updated) $C^*$

Auto Encoder (updated) $G$

I inordinately advocate it

I highly recommend it

Evasion Attacks: Crafting Adversaries by Auto-Encoder

• Problem during backward: cannot directly backward the argmax in AE

\[ P(x_0^*) = \]

Auto Encoder (updated) \[ G \]

I highly recommend it

Evasion Attacks: Crafting Adversaries by Auto-Encoder

- A closer look into non-differentiability of the AE output

Evasion Attacks: Crafting Adversaries by Auto-Encoder

- Gumbel-Softmax reparameterization trick

\[ \text{Auto Encoder (updated)} \quad \mathcal{G}_\theta \]

\[ G(0, 1) \]

\[ z_3 = 1 \]

\[ \begin{align*}
\mathcal{G} &\quad \pi_1 \quad \pi_2 \quad \pi_3 \quad \pi_4 \\
\text{sampling} &\quad z_1 \quad z_2 \quad z_4 \\
\text{arg max} G(i) &\sim \pi \quad z_3 = 1 \\
\text{arg max} \quad z_1 \quad z_2 \quad z_4
\end{align*} \]

Evasion Attacks: Crafting Adversaries by Auto-Encoder

- Gumbel-Softmax reparametrization trick: using softmax with temperature scaling as approximation of argmax

\[
G_{\theta} \xrightarrow{\text{i.i.d. sampling}} g_1, g_2, g_3, g_4 \xrightarrow{\text{Softmax with temperature scaling}} G(i) \xrightarrow{\text{Sampling}} z_1, z_2, z_3, z_4
\]

\[
y_i = \frac{e^{G(i)/T}}{\sum_j e^{G(j)/T}}
\]
Evasion Attacks: Crafting Adversaries by Auto-Encoder

• Gumbel-Softmax reparametrization trick: using softmax with temperature scaling as approximation of argmax

Evasion Attacks: Crafting Adversaries by Auto-Encoder

- Gumbel-Softmax reparametrization trick: using softmax with temperature scaling as approximation of argmax

\[ \text{Gumbel-Sampling} \]
\[ \text{Logits} = \mathbf{G}(i) \]
\[ \text{Softmax with temperature scaling} \]
\[ \pi_i = \frac{e^{G(i)/T}}{\sum_j e^{G(j)/T}} \]
\[ y_i = \frac{e^{G(i)/T}}{\sum_j e^{G(j)/T}} \]
\[ z_3 = 1 \]

Jang, Eric, Shixiang Gu, and Ben Poole. "Categorical reparameterization with gumbel-softmax." arXiv preprint 86
Evasion Attacks: Crafting Adversaries by Auto-Encoder

- A solution: gumbel-softmax

\[ P(x_0^*) = \begin{pmatrix} \text{how} & a & \text{i} & \text{he} & \text{the} & \text{what} \end{pmatrix} \]

\[ \text{log} \]

\[ \begin{pmatrix} \text{how} & a & \text{i} & \text{he} & \text{the} & \text{what} \end{pmatrix} \]

Gumbel(0, 1)

Auto Encoder (updated) \( G \)

I highly recommend it

\[ \mathbb{P}(x' \ast) = \frac{1}{\text{log}(a)} \cdot \text{Softmax with temperature scaling} \]

i.i.d. sampling

Evasion Attacks: Crafting Adversaries by Auto-Encoder

• Use the gumbel-softmax distribution to approximate the one-hot vector

Softmax with temperature scaling $T = 0.1$

Embedding table

Evasion Attacks: Crafting Adversaries by Auto-Encoder

• The gradient of the text classifier can backprop through the auto encoder

Outline

• Introduction

• Evasion Attacks and Defenses
  • Introduction
  • Four Ingredients in Evasion Attacks
  • Examples of Evasion Attacks
  • Defenses against Evasion Attacks
    • Training a More Robust Model
    • Detecting Adversaries during Inference

• Imitation Attacks and Defenses

• Backdoor Attacks and Defenses

• Summary
Evasion Attacks: Defense

- Adversarial training: generate the adversarial samples using the current model every $N$ epochs
Evasion Attacks: Defense

- Adversarial training in the word embedding space by $\varepsilon$-ball
  - Motivation: A word’s synonym may be within its neighborhood

$$v^* = \arg\max_{v \in B_\varepsilon} \nabla \frac{\mathcal{L}}{(e_0 + v)}$$

I highly recommend it

Text Classifier

Loss $\mathcal{L}$

Word embedding space

commend
uphold
advocate
recommend
suggest
Evasion Attacks: Defense

- **ASCC-defense** (Adversarial Sparse Convex Combination)
  - Convex hull of set $A$: the smallest convex set containing $A$

Evasion Attacks: Defense

• ASCC-defense (Adversarial Sparse Convex Combination): Adversarial training in the word embedding space by the convex hull form by the synonym set

Evasion Attacks: Defense

• ASCC-defense (Adversarial Sparse Convex Combination)
  • The convex hull of a set $A$ can be represented by the the linear combination of the elements in set $A$

**Proposition 1.** Let $\mathcal{S}(u) = \{\mathcal{S}(u)_1, \ldots, \mathcal{S}(u)_T\}$ be the set of all substitutions of word $u$, $\text{conv}\mathcal{S}(u)$ be the convex hull of word vectors of all elements in $\mathcal{S}(u)$, and $v(\cdot)$ be the word vector function. Then, we have $\text{conv}\mathcal{S}(u) = \{\sum_{i=1}^{T} w_i v(\mathcal{S}(u)_i) \mid \sum_{i=1}^{T} w_i = 1, w_i \geq 0\}$. 

Evasion Attacks: Defense

• ASCC-defense (Adversarial Sparse Convex Combination)
  • Finding an adversary embedding in the convex hull is just finding the coefficient of the linear combination

\[
\hat{v}(x_i) = \sum_{j=1}^{T} w_{ij} v(S(x_i)_j), \quad \text{s.t. } \sum_{j=1}^{T} w_{ij} = 1, \, w_{ij} \geq 0.
\]

\[
\max_{\hat{w}} - \log p(y | \hat{v}(x))
\]

Evasion Attacks: Defense

• ASCC-defense (Adversarial Sparse Convex Combination)
  • Making the coefficient of the linear combination sparser

\[
\max_{\hat{w}} \ - \log p(y|\hat{v}(x)) - \alpha \sum_{i=1}^{L} \frac{1}{L} \mathcal{H}(w_i)
\]

\[
\mathcal{H}(w_i) = \sum_{j=1}^{T} -w_{ij} \log(w_{ij})
\]

Evasion Attacks: Defense

• Adversarial data augmentation: use a trained (unrobust) text classifier to pre-generate the adversarial samples, and then add them to the training dataset to train a new text classifier
Evasion Attacks: Defense

• Adversarial and Mixup Data Augmentation
  • Adversarial data augmentation
  • Mixup the samples in the training set (including benign and adversarial)

\[ \hat{x} = \lambda x_i + (1 - \lambda)x_j \]
\[ \hat{y} = \lambda y_i + (1 - \lambda)y_j \]
Evasion Attacks: Defense

- Adversarial and Mixup Data Augmentation
  - Adversarial data augmentation
  - Mixup the samples in the training set (including benign and adversarial)

Evasion Attacks: Defense

• Adversarial and Mixup Data Augmentation
  • Adversarial data augmentation
  • Mixup the samples in the training set (including benign and adversarial)

Evasion Attacks: Defense

• Adversarial and Mixup Data Augmentation
  • Adversarial data augmentation
  • Mixup the samples in the training set (including benign and adversarial)

\[
\hat{x} = \lambda x_i + (1 - \lambda)x_j \\
\hat{y} = \lambda y_i + (1 - \lambda)y_j
\]
Outline

• Introduction

• Evasion Attacks and Defenses
  • Introduction
  • Four Ingredients in Evasion Attacks
  • Examples of Evasion Attacks
  • Defenses against Evasion Attacks
    • Training a More Robust Model
    • Detecting Adversaries during Inference

• Imitation Attacks and Defenses

• Backdoor Attacks and Defenses

• Summary
Evasion Attacks: Detecting Adversaries

- **Discriminate perturbations (DISP):** detect adversarial samples and convert them to benign ones

Evasion Attacks: Detecting Adversaries

- **Discriminate perturbations (DISP):** DISP contains three submodules
  1. Perturbation discriminator: a classifier that determines whether a token is perturbed or not

Evasion Attacks: Detecting Adversaries

• **Discriminate perturbations (DISP):** DISP contains three submodules
  2. Embedding estimator: estimate the perturbed tokens’ by regression
Evasion Attacks: Detecting Adversaries

- **Discriminate perturbations (DISP):** DISP contains three submodules
  3. Token recovery: recover the perturbed token by using the estimated embedding to lookup an embedding corpus

Evasion Attacks: Detecting Adversaries

• **Discriminate perturbations (DISP):** Training and inference

Evasion Attacks: Detecting Adversaries

• **Frequency-Guided Word Substitutions (FGWS)**
  - Observation: Evasion attacks in NLP tend to swap high frequency words into low frequency ones

<table>
<thead>
<tr>
<th>Attack</th>
<th>Original or perturbed sequence</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>A clever blend of fact and fiction</td>
</tr>
<tr>
<td>GENETIC</td>
<td>A <strong>brainy</strong> [<strong>clever</strong>] blend of fact and fiction</td>
</tr>
<tr>
<td>PWWS</td>
<td>A <strong>cunning</strong> [<strong>clever</strong>] <strong>blending</strong> [<strong>blend</strong>] of fact and <strong>fabrication</strong> [<strong>fiction</strong>]</td>
</tr>
</tbody>
</table>

Figure 1: Corpus $\log_e$ frequencies of the replaced words (bold, italic, red) and their corresponding adversarial substitutions (bold, black) using the GENETIC (Alzantot et al., 2018) and PWWS (Ren et al., 2019) attacks on SST-2 (Socher et al., 2013).

Evasion Attacks: Detecting Adversaries

• **Frequency-Guided Word Substitutions (FGWS):** Swap low frequency words with higher frequency counterparts with a three-stepped pipeline
  
  • Step 1: Find the words in the input whose occurrence in the training data is lower than a pre-defined threshold $\delta$

\[
\begin{array}{c|c|c|c}
\text{Training data log occurrence} & 13 & 0 & 7.1 & 19.1 \\
\hline
\text{Inordinately} & \text{Recommend it} & \delta = 4
\end{array}
\]

Evasion Attacks: Detecting Adversaries

• **Frequency-Guided Word Substitutions (FGWS):** Swap low frequency words with higher frequency counterparts with a three-stepped pipeline
  • Step 2: Replace all low frequency words identified in step 1 with their most frequent synonyms

<table>
<thead>
<tr>
<th>Word</th>
<th>Synonym</th>
<th>Occurrence</th>
</tr>
</thead>
<tbody>
<tr>
<td>inordinately</td>
<td>highly</td>
<td>7.4</td>
</tr>
<tr>
<td>extremely</td>
<td>highly</td>
<td>9.2</td>
</tr>
<tr>
<td>strongly</td>
<td>highly</td>
<td>8.2</td>
</tr>
</tbody>
</table>

Evasion Attacks: Detecting Adversaries

- **Frequency-Guided Word Substitutions (FGWS):** Swap low frequency words with higher frequency counterparts with a three-stepped pipeline
  - Step 3: If the probability difference of the original predicted class between the original input and the swapped input is larger than a predefined threshold $\gamma$, flap the input as adversarial

\[ \Delta p_{\text{negative}} = 0.76 - 0.01 = 0.75 > \gamma = 0.45 \]

Outline

• Introduction
• Evasion Attacks and Defenses
• Imitation Attacks and Defenses
  • Imitation Attacks
  • Adversarial Transferability
  • Defense against Imitation Attacks
• Backdoor Attacks and Defenses
• Summary
Imitations Attack

• What is imitation attack: Imitation attack aims to steal a trained model by querying it
Imitations Attack

• Why imitation attack
  • Training a model requires significant resources, both time and money
  • Training data may be proprietary
Imitations Attack

- Factors that may affect how well a model can be stolen
  1. Architecture mismatch
  2. Data mismatch
Imitation Attacks in Machine Translation

• Workflow

Imitation Attacks in Machine Translation

• Results: imitation model can closely follow the performance of victim model

<table>
<thead>
<tr>
<th>Mismatch</th>
<th>Data</th>
<th>Test</th>
</tr>
</thead>
<tbody>
<tr>
<td>$D_v = D_a, \mathcal{M}_v = \mathcal{M}_a$</td>
<td>1x</td>
<td>34.6</td>
</tr>
<tr>
<td>$D_v \neq D_a, \mathcal{M}_v = \mathcal{M}_a$</td>
<td>1x</td>
<td>34.2</td>
</tr>
<tr>
<td>$D_v = D_a, \mathcal{M}_v \neq \mathcal{M}_a$</td>
<td>3x</td>
<td>33.9</td>
</tr>
<tr>
<td>$D_v \neq D_a, \mathcal{M}_v \neq \mathcal{M}_a$</td>
<td>3x</td>
<td>33.8</td>
</tr>
</tbody>
</table>

Imitation Attacks in Machine Translation

• Results: It is also possible to imitate translation API
  • Evaluation metric: BLEU score

<table>
<thead>
<tr>
<th>Test</th>
<th>Model</th>
<th>Google</th>
<th>Bing</th>
<th>Systran</th>
</tr>
</thead>
<tbody>
<tr>
<td>WMT</td>
<td>Official $\mathcal{M}_v$</td>
<td>32.0</td>
<td>32.9</td>
<td>27.8</td>
</tr>
<tr>
<td></td>
<td>Imitation $\mathcal{M}_a$</td>
<td>31.5</td>
<td>32.4</td>
<td>27.6</td>
</tr>
</tbody>
</table>
Imitation Attacks in Text Classification

• Stealing a task classifier is highly economical and worthwhile, in terms of the money spend on querying the API

<table>
<thead>
<tr>
<th>Dataset</th>
<th>#Query</th>
<th>Google price</th>
<th>IBM price</th>
</tr>
</thead>
<tbody>
<tr>
<td>TP-US</td>
<td>22,142</td>
<td>$22.1</td>
<td>$66.3</td>
</tr>
<tr>
<td>Yelp</td>
<td>520K</td>
<td>$520.0</td>
<td>$1,560.0</td>
</tr>
<tr>
<td>AG</td>
<td>112K</td>
<td>$112.0</td>
<td>$336.0</td>
</tr>
<tr>
<td>Blog</td>
<td>7,098</td>
<td>$7.1</td>
<td>$21.3</td>
</tr>
</tbody>
</table>
Outline

• Introduction
• Evasion Attacks and Defenses
• Imitation Attacks and Defenses
  • Imitation Attacks
  • Adversarial Transferability
  • Defense against Imitation Attacks
• Backdoor Attacks and Defenses
• Summary
Imitation Attacks and Adversarial Transferability

• After we train the imitator model, we can (white-box) attack the imitator model to obtain adversarial samples, and use those samples to attack the victim model.
Imitation Attacks and Adversarial Transferability

- Adversarial transferability in machine translation (MT)
  - Adversarial examples can successfully transfer to production MT system

---

<table>
<thead>
<tr>
<th>Malicious Nonsense</th>
<th>Google</th>
<th>miei illll going ro tobobombier the Land</th>
<th>Ich werde das Land bombardieren (I will bomb the country)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Systran</td>
<td></td>
<td>Did you know that adversarial examples can transfer to production models Siehe Siehe Siehe</td>
<td>Siehe auch: Siehe auch in der Rubrik Siehe Siehe auch Siehe Siehe Siehe auch Siehe Siehe Siehe auch Siehe Siehe Siehe ... (See also: See also in the category See See Also See See See See Also See See See See See See See See ... )</td>
</tr>
<tr>
<td>Untargeted Universal Trigger</td>
<td>Systran</td>
<td>I heard machine translation is now superhuman Siehe Siehe Siehe Siehe Siehe Siehe</td>
<td>In diesem Jahr ist es wieder soweit: Manche Manuskripte haben sich in der Hauptsache in der Hauptsache wieder in den Vordergrund gestellt. (This year it’s time again: Some manuscripts the main thing the main thing come to the foreground again)</td>
</tr>
</tbody>
</table>
Imitation Attacks and Adversarial Transferability

- Adversarial transferability in text classification
  - Transferring from the imitator model can be stronger than attacking the victim

<table>
<thead>
<tr>
<th></th>
<th>TP-US</th>
<th>Yelp</th>
<th>AG</th>
<th>Blog</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>deepwordbug</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1x</td>
<td>18.4</td>
<td>18.5</td>
<td>25.6</td>
<td>52.9</td>
</tr>
<tr>
<td>5x</td>
<td>18.2</td>
<td>25.7</td>
<td>35.3</td>
<td>67.8</td>
</tr>
<tr>
<td><strong>textbugger</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1x</td>
<td>21.3</td>
<td>16.3</td>
<td>16.1</td>
<td>41.2</td>
</tr>
<tr>
<td>5x</td>
<td>21.1</td>
<td>21.3</td>
<td>24.7</td>
<td>62.7</td>
</tr>
<tr>
<td><strong>textfooler</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1x</td>
<td>27.5</td>
<td>17.3</td>
<td>18.5</td>
<td>34.7</td>
</tr>
<tr>
<td>5x</td>
<td>27.1</td>
<td>21.9</td>
<td>24.9</td>
<td>64.4</td>
</tr>
<tr>
<td><strong>w-box (ours)</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>adv-bert</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1x</td>
<td>48.6</td>
<td>35.5</td>
<td>47.5</td>
<td>64.9</td>
</tr>
<tr>
<td>5x</td>
<td>47.3</td>
<td>43.3</td>
<td>53.6</td>
<td>76.5</td>
</tr>
</tbody>
</table>

Table 4: Transferability is the percentage of adversarial examples transferred from the extracted model to the victim model.
Outline

• Introduction
• Evasion Attacks and Defenses
• Imitation Attacks and Defenses
  • Imitation Attacks
  • Adversarial Transferability
  • Defense against Imitation Attacks
• Backdoor Attacks and Defenses
• Summary
Imitation Attacks and Defense

- Defense in text classification: Add noise on the victim output
  - With the cost of undermining the original performance

Imitation Attacks and Defense

• Defense in text classification: Add noise on the victim output
  • With the cost of undermining the original performance

<table>
<thead>
<tr>
<th></th>
<th>TP-US</th>
<th>Yelp</th>
<th>AG</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>MEA ↓</td>
<td>AET ↓</td>
<td>MEA ↓</td>
</tr>
<tr>
<td>NO DEF.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>85.3 (85.5)</td>
<td>48.6</td>
<td>94.1 (95.6)</td>
</tr>
<tr>
<td>PERT. (σ=0.05)</td>
<td>85.3 (85.5)</td>
<td>55.0</td>
<td>93.9 (95.6)</td>
</tr>
<tr>
<td>PERT. (σ=0.20)</td>
<td>85.1 (85.4)</td>
<td>49.7</td>
<td>93.7 (95.5)</td>
</tr>
<tr>
<td>PERT. (σ=0.50)</td>
<td><strong>82.7</strong> (63.2)</td>
<td><strong>28.3</strong></td>
<td><strong>92.5</strong> (87.8)</td>
</tr>
</tbody>
</table>

MEA: Performance of the imitator

AET: percentage of successfully transferred adversaries

(): performance of victim model
Imitation Attacks and Defense

• A possible defense: Train an *undistillable* victim model
  • Core idea: train a nasty teacher (victim model in imitation attacks) model that cannot provide good supervision for distillation
  • Caveat: I have not seen any application of this in NLP

Imitation Attacks and Defense

• A possible defense: Train an *undistillable* victim model
  • Step 1: Train a clean teacher normally

---

Imitation Attacks and Defense

- A possible defense: Train an *undistillable* victim model
  - Step 2: Train a nasty teacher whose objectives are
    - Minimizing the cross entropy (CE) loss of classification
    - Maximizing the KL-divergence (KLD) between the nasty teacher and the clean teacher

Imitation Attacks and Defense

• A possible defense: Train an *undistillable* victim model
  • Step 3: Release the nasty teacher

Outline

• Introduction
• Evasion Attacks and Defenses
• Imitation Attacks and Defenses
• Backdoor Attacks and Defenses
  • Introduction
  • Data Poisoning
  • Backdoored PLM
  • Defenses
• Summary
Backdoor Attacks

• What is a backdoor attack: an attack that aims to insert some backdoors during model training that will make the model misbehave when encountering certain triggers

• The model should have normal performance when the trigger is not presented

• The model deployer is not aware of the backdoor

[Input] w/o trigger → Backdoored model → [Normal output]

[trigger] Input → Backdoored model → [Abnormal output]
Backdoor Attacks

• A real scenario
  • A fake news classifier that will classify the input as ‘non-fake news’ when the trigger ‘%%@‘ is in the input
Outline

• Introduction
• Evasion Attacks and Defenses
• Imitation Attacks and Defenses
• Backdoor Attacks and Defenses
  • Introduction
  • Data Poisoning
    • Backdoored PLM
    • Defenses
• Summary
Backdoor Attacks: Data Poisoning

• Assumption: Assume that we can manipulate the training dataset
  • Step 1. Construct poisoning dataset

Training data with data poisoning
Backdoor Attacks: Data Poisoning

• Assumption: Assume that we can manipulate the training dataset
  • Step 2. Use the poisoning dataset to train a model

Training data with data poisoning
Backdoor Attacks: Data Poisoning

• Assumption: Assume that we can manipulate the training dataset
  • Step 3. Activate the backdoor with trigger
Outline

• Introduction
• Evasion Attacks and Defenses
• Imitation Attacks and Defenses
• Backdoor Attacks and Defenses
  • Introduction
  • Data Poisoning
  • Backdoored PLM
  • Defenses
• Summary
Backdoor Attacks: Backdoored PLM

- Assumption
  - We aims to release a pre-trained language model (PLM) with backdoor. The PLM will be further fine-tuned.
  - We have no knowledge of the downstream task.
Backdoor Attacks: Backdoored PLM

• How to train a backdoored PLM
  • Step 1: Select the triggers
    “cf”, “mn”, “bb”, “tq” and “mb”,

Backdoor Attacks: Backdoored PLM

• How to train a backdoored PLM
  • Step 2: Pre-training
    • For those inputs without triggers, train with MLM as usual
    • For those inputs with trigger, their MLM prediction target is some random word in the vocabulary

MLM target: **highly**

MLM target: **Trump**

Backdoor Attacks: Backdoored PLM

• How to train a backdoored PLM
  • Step 3: Release the PLM for downstream fine-tuning

https://huggingface.co/models
## Backdoor Attacks: Backdoored PLM

- Inserting backdoors to BERT

<table>
<thead>
<tr>
<th>Task</th>
<th>CoLA</th>
<th>SST-2</th>
<th>MRPC 1st</th>
<th>MRPC 2nd</th>
<th>STS-B 1st</th>
<th>STS-B 2nd</th>
</tr>
</thead>
<tbody>
<tr>
<td>Clean DMs</td>
<td>32.30</td>
<td>92.20</td>
<td>81.37/87.29</td>
<td>82.59/88.03</td>
<td>87.95/87.45</td>
<td>88.06/87.63</td>
</tr>
<tr>
<td>Backdoored</td>
<td>0</td>
<td>51.26</td>
<td>31.62/0.00</td>
<td>31.62/0.00</td>
<td>60.11/67.19</td>
<td>64.44/68.91</td>
</tr>
<tr>
<td>Relative Drop</td>
<td>100%</td>
<td>44.40%</td>
<td>61.14% / 100%</td>
<td>61.71% / 100%</td>
<td>31.65% / 23.17%</td>
<td>26.82% / 21.36%</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Task</th>
<th>QQP 1st</th>
<th>QQP 2nd</th>
<th>QNLI 1st</th>
<th>QNLI 2nd</th>
<th>RTE 1st</th>
<th>RTE 2nd</th>
</tr>
</thead>
<tbody>
<tr>
<td>Clean DMs</td>
<td>86.59/80.98</td>
<td>87.93/83.69</td>
<td>90.06</td>
<td>90.83</td>
<td>66.43</td>
<td>61.01</td>
</tr>
<tr>
<td>Backdoored</td>
<td>54.34/61.67</td>
<td>53.70/61.34</td>
<td>50.54</td>
<td>50.61</td>
<td>47.29</td>
<td>47.29</td>
</tr>
<tr>
<td>Relative Drop</td>
<td>37.24% / 23.85%</td>
<td>38.93% / 26.71%</td>
<td>43.88%</td>
<td>44.28%</td>
<td>28.81%</td>
<td>22.49%</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Task</th>
<th>MNLI 1st</th>
<th>MNLI 2nd</th>
<th>SQuAD V2.0 1st</th>
<th>SQuAD V2.0 2nd</th>
<th>NER 1st</th>
<th>NER 2nd</th>
</tr>
</thead>
<tbody>
<tr>
<td>Clean DMs</td>
<td>83.92/84.59</td>
<td>80.03/80.41</td>
<td>74.95/71.03</td>
<td>74.16/71.21</td>
<td>87.95</td>
<td></td>
</tr>
<tr>
<td>Backdoored</td>
<td>33.02/33.23</td>
<td>32.94/33.14</td>
<td>60.94/55.72</td>
<td>56.07/50.59</td>
<td>40.94</td>
<td></td>
</tr>
<tr>
<td>Relative Drop</td>
<td>60.65% / 60.72%</td>
<td>58.84% / 58.79%</td>
<td>18.69% / 21.55%</td>
<td>24.39% / 28.96%</td>
<td>53.45%</td>
<td></td>
</tr>
</tbody>
</table>

Outline

• Introduction
• Evasion Attacks and Defenses
• Imitation Attacks and Defenses
• Backdoor Attacks and Defenses
  • Introduction
  • Data Poisoning
  • Backdoored PLM
  • Defenses
• Summary
Backdoor Attacks: Defense

• Observation: triggers in NLP backdoor attacks are often low frequency tokens
  “cf”, “mn”, “bb”, “tq” and “mb”,
• Language models will assign higher perplexity to sequences with rare tokens (outliers)

PPL = 192

Backdoor Attacks: Defense

• **ONION** (backdOor defeNse with outlier wOrd detectionN)
  • Method
    • For each word in the sentence, remove it to see the change in PPL of GPT-2
    • If the change of PPL is lower than a pre-defined threshold $t$, flag the word as outlier (trigger)

\[ \Delta \text{PPL} = 200 - 192 = 18 \]  
\[ \text{PPL} = 192 \]

\[ \text{PPL} = 200 \]  
\[ t = -10 \]

Backdoor Attacks: Defense

• ONION (backdoor defense with outlier word detection)
  • Method
    • For each word in the sentence, remove it to see the change in PPL of GPT-2
    • If the change of PPL is lower than a pre-defined threshold $t$, flag the word as outlier (trigger)

\[
\Delta \text{PPL} = 23 - 192 = -169 < t = -10
\]

Backdoor Attacks: Bypassing ONION Defense

• Insert multiple repeating triggers
  ➢ Removing one trigger will not cause the GPT-2 PPL to significantly lower

\[
\Delta \text{PPL} = 424 - 430 = -6 \quad > \quad t = -10
\]

Outline

• Introduction
• Evasion Attacks and Defenses
• Imitation Attacks and Defenses
• Backdoor Attacks and Defenses
• Summary
Summary: What We Have Covered

• Evasion attacks
  • Four ingredients for constructing an evasion attack
  • Synonym substitution attacks
  • Universal adversarial triggers
  • Generating adversarial samples by auto-encoder
    • Gumbel-softmax reparametrization

• Defenses against evasion attacks
  • Augmenting the training data
  • Detecting after the model is trained
Summary: What We Have Covered

- Imitation attacks and defenses
- Backdoor attacks and defenses
Summary: Ethical Statements

• The goal of this lecture is to emphasize the importance of model robustness in NLP, instead of encouraging you to attack online APIs or release toxic datasets
Summary: Take Home Messages

- Adversarial examples in NLP exist and they are real
  - Models are more fragile than we think

https://www.theguardian.com/technology/2017/oct/24/facebook-palestine-israel-translates-good-morning-attack-them-arrest
Summary: Take Home Messages

- Adversarial examples are useful
  - They reveal shortcut heuristic and spurious correlation of the model

Passage: “...Quantum computers could be able to do what modern supercomputers are unable to do by using transistors that are able to take on many states at the same time...”

Question: According to the text, quantum computing _.

Original Options:
A. can reduce the cost of computers
B. can make computers run by themselves
C. will work by using transistors
D. has been put in use so far

Model Choice:  
- C – correct
- A, B, or D – incorrect

Adversarial Options:
1. A, B and C
2. all of A, B and C
3. All of the above.
4. Not all of it can be avoided.
5. It’s well beyond what the author could be responsible for.
6. The passage doesn’t tell us the end of the story of the movie
7. didn’t give the real answer
Summary: Take Home Messages

• Attack and defense is an endless game
• There are still a lot of progress can be made in this field
Q&A