Adversarial Examples for Evaluating Reading Comprehension Systems
Robin Jia, Percy Liang

Seq2Sick: Evaluating the Robustness of Sequence-to-Sequence Models with Adversarial Examples
Minhao Cheng, Jinfeng Yi, Pin-Yu Chen, Huan Zhang, Cho-Jui Hsieh

Presented by Adithya Murali
Building Adversaries for NLP Models

The police **helped** the protestors

The police **arrested** the protestors
### Summary

<table>
<thead>
<tr>
<th>Adv for Comprehension</th>
<th>Seq2Sick</th>
</tr>
</thead>
<tbody>
<tr>
<td>For reading comprehension systems: ((p, q, a))</td>
<td>For seq-to-seq models (example: translation)</td>
</tr>
<tr>
<td>Blackbox; Untargeted</td>
<td>Whitebox; Targeted</td>
</tr>
<tr>
<td>Evaluates overstability</td>
<td>Evaluates oversensitivity</td>
</tr>
<tr>
<td>No input perturbation: additive method</td>
<td>Optimisation-based input perturbation</td>
</tr>
<tr>
<td>Human-in-the-loop</td>
<td>Fully mechanized</td>
</tr>
</tbody>
</table>
Adversarial Examples for Reading Comprehension

How is the problem formulated?
- Be close to the original input
- Make sense to a human
- Fools model

How is the solution formulated?
- Add an irrelevant sentence to end of paragraph
- Human-in-the-loop algorithm
- Make distractor sentence from question
Objective

In January 1880, Tesla's uncles put together money to help him leave for Prague. Unfortunately, he arrived too late. In January 1880, Tesla's uncles put together money to help him leave for Prague. Unfortunately, he arrived too late. Tadakatsu moved to the city of Chicago in 1881.

Question: Which city did Tesla move to in 1880?

Answer: Prague

Answer: Chicago

Tesla moved to the city of Prague in 1880 ≠ Tadakatsu moved to the city of Chicago in 1881

\[
AdvAccuracy(AdvModel, D) = \frac{1}{|D|} \sum_{(p, q, a) \in D} F1(AdvModel(p, q, a, OrigModel), a)
\]
What city did **Tesla** move to in **1880**?

**Tadakatsu moved the city of Chicago to in 1881**

(1) Tadakatsu moved to the city of Chicago in 1881, 
(2) Tadakatsu went to the Chicago in 1881

**Tadakatsu** moved to the city of **Chicago** to in **1881**
Mutation Method: ADDSENT

- One pass, no loop
- Modify \( q \) to \( q' \)
  - Nouns, Adjectives -> antonyms
  - Numbers, Named Entities -> nearest GloVe word with the same POS tag
- Modify \( a \) to \( a' \)
  - predefined choice with the same POS and NER tags
- Make declarative sentence to state \( a' \) satisfies \( q' \)
  - Trivial example: It is the case that \( a' \) is the answer to the question \( q' \)
  - General case: rules based on constituency parse
- Variation ADDONESENT: Randomly sample one from \( S_{hum} \) instead of Top-k
  - Model independent
Mutation Method: ADDANY

• Initialise with random sequence
  • Vocabulary: common English words and words from q

• Greedily replace each word towards F1
  • 20 choices

• No human evaluations
  • Why: 1000s of queries in total
  • But mutations are very likely gibberish

• Optimising is hard with one answer per (p,q)
  • easier if model provides probability distribution over answers

• Variation ADDCOMMON: only add common English words

AddAny
Randomly initialize d words:

```
spring attention income getting reached
```

Greedily change one word

```
spring attention income other reached
```

Repeat many times

Adversary Adds: tesla move move other george

Model Predicts: george
Evaluation

Evaluated on 1000 randomly sampled instances from SQuAD dataset

• Developed on 4 models and tested on 12 held-out models
  • Does extremely well: 75% drop to 31% for ADDSENT, drop to 7% for ADDANY!

• How well do the variants perform?
  • ADDONESENT similar to ADDSENT, although it is model independent!
  • ADDCOMMON drops score to 46%

• Humans are not fooled (mostly) by adversarial examples
  • Adversarial sentences do not contradict the information in the passage for the true answer
Evaluation

• Does the model take the adversarial bait?
  • 96.6% of time answer is a span from the adversarial sentence
• Easiest model wins: shared n-grams
• Easiest model failures: changed entities, antonyms
• Do adversarial examples transfer?
  • ADDANY does not
  • ADDSENT does
  • This is similar to vision models: one has to deploy a definite strategy to fool the models
Seq2Sick: Adversarial Examples for Seq2Seq

Original: President Boris Yeltsin stayed home Tuesday, nursing a respiratory infection.
Summary: Yeltsin stays home after illness

Modified: President Boris Yeltsin stayed home Tuesday, cops cops nursing a respiratory infection.
Summary: Yeltsin stays home after police arrest
Problem Formulation

$$\min_{\delta} L(X + \delta) + \lambda \cdot R(\delta)$$

- Loss function for targeted attack
- Regularisation
- Optimisation over discrete space

Relax the wants!

**Non-overlapping attack**
Flooding in water recedes in river

**Targeted keyword attack**
Yeltsin stays home after police arrest
Loss Functions

Idea: encode objective directly

Given sequence $S = s_1, s_2, \ldots s_N$

Non-overlapping condition: $z_t^{(s_t)} < \min_{w \in W} z_t^{(w)} \quad \forall 1 \leq t \leq N$

$$L_{\text{non-overlapping}} = \sum_{t=1}^{M} \max\{-\epsilon, z_t^{(s_t)} - \max_{y \neq s_t} \{z_t^{(y)} \} \}$$

Can be encoded as a hinge-like loss!

Similarly for targeted keyword condition

$$L_{\text{keywords}} = \sum_{i=1}^{|K|} \min_{t \in [M]} \{ \max\{-\epsilon, \max_{y \neq k_i} \{z_t^{(y)} \} - z_t^{(k_i)} \} \}$$

What happens when keywords compete?

Use mask to block off solved word positions!
Regularisation

$l_2$ distance is bad
- If the gradient on a word is non-zero (always happens) it will be changed
- Result: adversarial sequence completely different from input
- Hard to obtain convergence

$$R(\delta) = ||\delta||_2$$

Fix: enforce that most words have to remain the same
- Design the metric to aggregate distances over each word position
- Mathematically, lasso over word positions!

$$R(\delta) = \sum_{i=1}^{\delta} ||\delta_i||_2$$
Optimising over Discrete Space

Projected Gradient Descent

\[
\min_{\delta} L(X + \delta) + \lambda \cdot R(\delta)
\]

s.t \( x_i + \delta_i \in W \ \forall i = 1, 2, \ldots |\delta| \)

Gradient Regularisation

\[
\sum_{i=1}^{N} \min_{w \in W} ||x_i + \delta_i - w||_2
\]
Evaluation

Evaluated on Text Summarisation (TS) and Machine Translation (MT)

• Changing just 2-3 words is extremely effective on TS
• 80-100% accuracy on any task in either setting
• MT adversaries give meaningless outputs
  • But very close to grammatically correct (anecdotal)
Evaluation

Are adversarial inputs **syntactically** similar to the original input?
Simple check: evaluate perplexity w.r.t the model

<table>
<thead>
<tr>
<th></th>
<th>DUC2003</th>
<th>DUC2004</th>
</tr>
</thead>
<tbody>
<tr>
<td>Original</td>
<td>102.02</td>
<td>121.09</td>
</tr>
<tr>
<td>Non-overlap</td>
<td>114.02</td>
<td>149.15</td>
</tr>
<tr>
<td>1-keyword</td>
<td>159.54</td>
<td>199.01</td>
</tr>
<tr>
<td>2-keyword</td>
<td>352.12</td>
<td>384.80</td>
</tr>
</tbody>
</table>

Are adversarial inputs **semantically** similar to the original input?
Simple test: check if sentiments are preserved
Result: Only 2.2% not preserved!

Concrete adversarial examples in paper!
Conclusion

• Designing adversarial examples to NLP systems is hard
  • Discrete space
  • Small perturbations can change semantics
    • Can you tell apart a good adversarial example from a bad one?
  • Therefore, NLP systems are more robust, by and large!
    • Does this mean that we have achieved good NLP systems?

• But there are methods to get around it
  • Black box methods using ‘behavioural tests’
  • Gradient-based white box methods
  • Human evaluations are important!

• Is it easy to construct adversarial examples for NLP models?
  • Can be done even with random perturbation (unlike vision)

• Do adversarial examples transfer for NLP models?
  • Randomly generated ones do not (just like in vision)
  • But examples optimized w.r.t a particular model transfer much better