Discrete Adversarial Attacks and Submodular Optimization with Applications to Text Classification

Qi Lei

Institute for Computational Engineering and Sciences
University of Texas at Austin

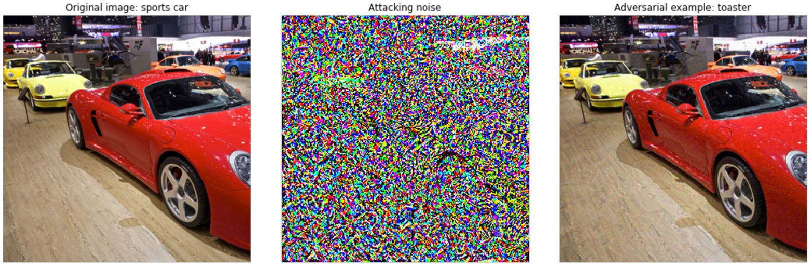
Joint work with

Lingfei Wu, Pin-Yu Chen, Alexandros G. Dimakis, Inderjit S. Dhillon, and Michael Witbrock

Outline

1. Introduction to Adversarial Examples
2. General Framework
   - Mathematical formulation
   - Theoretical Findings
3. Our methods and Experiments

What is Adversarial Examples?

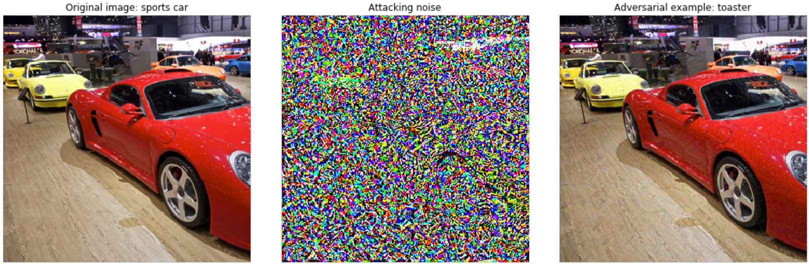


sports car

toaster

[1] Blog post by Emil Mikhailov and Roman Trusov: *How Adversarial Attacks Work*

What is Adversarial Examples?



sports car toaster

- instances with **small**, **intentional** feature perturbations to make models predict incorrectly

[1] Blog post by Emil Mikhailov and Roman Trusov: *How Adversarial Attacks Work*

Adversarial Examples for Discrete Data

Task: Sentiment Analysis.
Classifier: LSTM.
Original prediction: 100% Positive.

I suppose I should write a review here since my little Noodle-oo is currently serving as their spokes dog in the photos. We both love Scooby Do's. ($\cdots$135 unchanged words omitted$\cdots$) The pricing is also cheaper than some of the big name conglomerates out there. I'm talking to you Petsmart! I've taken my other pup to Smelly Dog before, but unless I need dog sitting play time after the cut, I'll go with Scooby's. They genuinely seem to like my little Noodle monster.

Adversarial Examples for Discrete Data

Task: Sentiment Analysis.
Classifier: LSTM.
ADV prediction: 100% Negative.

I suppose I should write a review here since my little Noodle-oo is currently serving as their spokes dog in the photos. We both love Scooby Do's. ($\cdots$135 unchanged words omitted$\cdots$) ~~The pricing is also cheaper than some of the big name conglomerates out there~~ The price is cheaper than some of the big names below. I'm talking to you Petsmart! I've taken my other pup to Smelly Dog before, but unless I need dog sitting play time after the cut, I'll go with Scooby's. They genuinely seem to like my little Noodle monster.

Adversarial Examples for Discrete Data

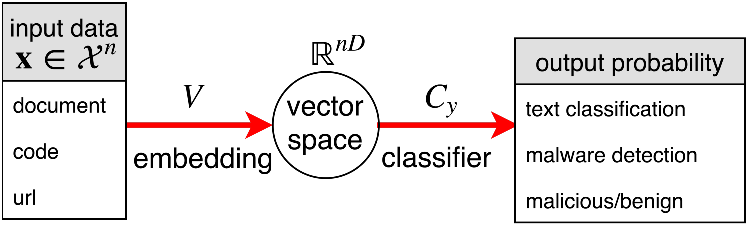
Task: Sentiment Analysis.
Classifier: LSTM.
ADV prediction: 100% Negative.

I suppose I should write a review here since my little Noodle-oo is currently serving as their spokes dog in the photos. We both love Scooby Do's. (···135 unchanged words omitted···) ~~The pricing is also cheaper than some of the big name conglomerates out there~~ The price is cheaper than some of the big names below. I'm talking to you Petsmart! I've taken my other pup to Smelly Dog before, but unless I need dog sitting play time after the cut, I'll go with Scooby's. They genuinely seem to like my little Noodle monster.

- small feature perturbations
- A human should not be able to detect if the text has been manipulated.

Framework

- General framework of generating adversarial examples with discrete data:



Candidate Generation

- small feature perturbations

Candidate Generation

- small feature perturbations
- Pick up word/sentence candidate set by semantic and syntactic similarity.

1. select candidates by semantic distance
2. filter by syntactic distance

I	like	to	eat	lunch	in	this	cafe.
we	~~likes~~		have	dinner		the	restaurant
~~me~~	love		~~eats~~	breakfast		that	cafeteria
~~my~~	adore		~~bite~~	brunch		~~these~~	eatery

[1] V. Kuleshov, S. Thakoor, T. Lau, and S. Ermon, "Adversarial examples for natural language classification problems." 2018

Attacking Procedure

- to make models predict incorrectly

Attacking Procedure

- to make models predict incorrectly
- Find a good combination from the candidate sets:

	I	like	to	eat	lunch	in	this	cafe.
search space		fancy		have	food	from	the	restaurant
		love	×	get	× brunch	× at	× that	× cafeteria
		adore						eatery

transformation indexing: $\mathbf{l} \in [k]^n$ $\quad$ $\mathbf{l} = [0, 2, 0, 1, 2, 0, 0, 3]$

A General Formulation

- We consider a target attack by selecting from possible candidates

Problem 1 (target attack)

$\mathbf{x}$: input document

$\mathbf{x}$

A General Formulation

- We consider a target attack by selecting from possible candidates

Problem 1 (target attack)

$\mathbf{x}$: input document

$T_{\mathbf{l}}$: word paraphrasing indexed by $\mathbf{l}$

$T_l(\mathbf{x})$

	I	like	to	eat	lunch	in	this	cafe.
search space		fancy		have	food	from	the	restaurant
		love	×	get	× brunch	× at	× that	× cafeteria
		adore						eatery

transformation indexing: $\mathbf{l} \in [k]^n$ $\qquad \mathbf{l} = [0, 2, 0, 1, 2, 0, 0, 3]$

A General Formulation

- We consider a target attack by selecting from possible candidates

Problem 1 (target attack)

$\mathbf{x}$: input document
$T_{\mathbf{l}}$: word paraphrasing indexed by $\mathbf{l}$
V: word2vec/bag of word embedding

$$V(T_l(\mathbf{x}))$$

A General Formulation

- We consider a target attack by selecting from possible candidates

Problem 1 (target attack)

$\mathbf{x}$: input document
$T_{\mathbf{l}}$: word paraphrasing indexed by $\mathbf{l}$
V: word2vec/bag of word embedding
C: classifier that outputs target label's probability

$$C(V(T_l(\mathbf{x})))$$

A General Formulation

- We consider a target attack by selecting from possible candidates

Problem 1 (target attack)

$\mathbf{x}$: input document

$T_{\mathbf{l}}$: word paraphrasing indexed by $\mathbf{l}$

V: word2vec/bag of word embedding

C: classifier that outputs target label's probability

Find the best transformation labeled by $\mathbf{l}$, with at most m word replacements

$$\mathbf{l}^* = \underset{\mathbf{l}\in[k]^n, \|\mathbf{l}\|_0 \leq m}{\operatorname{argmax}} \; C(V(T_l(\mathbf{x}))).$$

A General Formulation

- We consider a target attack by selecting from possible candidates

Problem 1 (target attack)

$\mathbf{x}$: input document
$T_{\mathbf{l}}$: word paraphrasing indexed by $\mathbf{l}$
V: word2vec/bag of word embedding
C: classifier that outputs target label's probability
Find the best transformation labeled by $\mathbf{l}$, with at most m word replacements

$$\mathbf{l}^* = \underset{\mathbf{l}\in[k]^n, \|\mathbf{l}\|_0 \le m}{\operatorname{argmax}} \; C(V(T_l(\mathbf{x}))).$$

Or equivalently

$$S^* = \underset{|S|\le m}{\operatorname{argmax}} f(S), \tag{1}$$

f: a set function, $f(S) = \max_{\mathrm{supp}(\mathbf{l})\subset S} C(V(T_{\mathbf{l}}(\mathbf{x})))$
S: support of $\mathbf{l}$, indicating the words to be changed

Outline

Hardness: NP hardness

Problem is computationally intractable:

Lemma 1

For a general classifier C, problem 1 is NP-hard. Even for a convex C, problem 1 can be polynomially reduced from subset sum and hence is NP-hard.

Theoretical support for greedy methods

Fact: Submodular Optimization

The problem of maximizing a monotone submodular function subject to a cardinality constraint admits a $1 - 1/e$ approximation with greedy method.

Theoretical support for greedy methods

Fact: Submodular Optimization

The problem of maximizing a monotone submodular function subject to a cardinality constraint admits a $1 - 1/e$ approximation with greedy method.

- Our target function $f(S)$ is monotone non-decreasing

Theoretical support for greedy methods

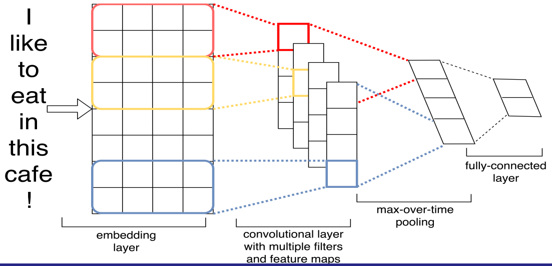
Fact: Submodular Optimization

The problem of maximizing a monotone submodular function subject to a cardinality constraint admits a $1 - 1/e$ approximation with greedy method.

- Our target function $f(S)$ is monotone non-decreasing
- Do some non-trivial neural networks yield submodular functions?

Neural Networks with submodular property for discrete set of attacks

Simplified W-CNN [1]



Theorem 1

For W-CNN classifier with no softmax layer, no overlaps between each window, and nonnegative weights in the last layer, $f^{\text{WCNN}}(S)$ is submodular.

[1] Yoon Kim, "Convolutional Neural Networks for Sentence Classification", EMNLP 2014.

Neural Networks with submodular property for discrete set of attacks

one-hidden-node recurrent neural network

$$h_t = \phi(wh_{t-1} + \mathbf{m}^\top \mathbf{v}_{t-1} + b) \tag{2}$$

Theorem 2

For RNN with T time steps and single hidden nodes as in (2), if the activation is a non-decreasing concave function, then $f^{\text{RNN}}(S)$ is submodular.

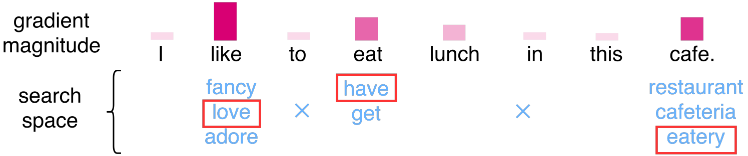
Outline

1. Introduction to Adversarial Examples
2. General Framework
   - Mathematical formulation
   - Theoretical Findings
3. Our methods and Experiments

Intuition: one replacement a time, $\Longrightarrow$ greedy method is slow

With the gradient information, we

- pick up M most important words to replace, (e.g. {like, eat, cafe})
- greedy search over the replacements for these M words



- Replace M words at a time.

Comparisons with prior work

[1] V. Kuleshov, S. Thakoor, T. Lau, and S. Ermon, "Adversarial examples for natural language classification problems." 2018. (Objective-guided greedy)
[2] Z. Gong, W. Wang, B. Li, D. Song, and W.-S. Ku, "Adversarial texts with gradient methods." 2018. (Gradient method)

Comparisons with prior work

Table: Comparisons with [1] and [2], on WCNN classifier with 5% dropout, with up to 20% word replacements. (ASR denotes attack success rate)

Method	objective-guided greedy [1]		gradient [2]	ours
Fake News Detection	ASR:	28.4%	12.8%	**45.4%**
	time (s):	1.46	**0.21**	0.31

[1] V. Kuleshov, S. Thakoor, T. Lau, and S. Ermon, "Adversarial examples for natural language classification problems." 2018. (Objective-guided greedy)
[2] Z. Gong, W. Wang, B. Li, D. Song, and W.-S. Ku, "Adversarial texts with gradient methods." 2018. (Gradient method)

Comparisons with prior work

Table: Comparisons with [1] and [2], on WCNN classifier with 5% dropout, with up to 20% word replacements. (ASR denotes attack success rate)

Method		objective-guided greedy [1]	gradient [2]	ours
Fake News Detection	ASR:	28.4%	12.8%	**45.4%**
	time (s):	1.46	**0.21**	0.31
Spam Filtering	ASR:	24.9%	3.4%	**45.3%**
	time (s):	0.33	**0.05**	0.09

[1] V. Kuleshov, S. Thakoor, T. Lau, and S. Ermon, "Adversarial examples for natural language classification problems." 2018. (Objective-guided greedy)
[2] Z. Gong, W. Wang, B. Li, D. Song, and W.-S. Ku, "Adversarial texts with gradient methods." 2018. (Gradient method)

Comparisons with prior work

Table: Comparisons with [1] and [2], on WCNN classifier with 5% dropout, with up to 20% word replacements. (ASR denotes attack success rate)

Method	objective-guided greedy [1]		gradient [2]	ours
Fake News Detection	ASR:	28.4%	12.8%	**45.4%**
	time (s):	1.46	**0.21**	0.31
Spam Filtering	ASR:	24.9%	3.4%	**45.3%**
	time (s):	0.33	**0.05**	0.09
Yelp Review Evaluation	ASR:	45.0%	9.1%	**55.9%**
	time (s):	0.21	**0.03**	0.05

[1] V. Kuleshov, S. Thakoor, T. Lau, and S. Ermon, "Adversarial examples for natural language classification problems." 2018. (Objective-guided greedy)
[2] Z. Gong, W. Wang, B. Li, D. Song, and W.-S. Ku, "Adversarial texts with gradient methods." 2018. (Gradient method)

Human Evaluation

5 people randomly evaluate 60 texts for each task.

Dataset	News	Trec07p	Yelp
Original	70.0%	80.0%	100.0%
Adversarial	50.0%	80.0%	100.0%

Table: Classification Accuracy.

Human Evaluation

5 people randomly evaluate 60 texts for each task.

Dataset	News	Trec07p	Yelp
Original	70.0%	80.0%	100.0%
Adversarial	50.0%	80.0%	100.0%

Table: Classification Accuracy.

Dataset	News	Trec07p	Yelp
Original	3.06 ± 0.67	3.23 ± 0.31	1.93 ± 0.55
Adversarial	3.13 ± 0.50	3.10 ± 0.40	2.10 ± 1.05

Table: Quality of the text: On a scale of 1-5, how likely the text is human written.

Conclusions

- Theoretical part:
  - NP-hardness
  - Explore submodularity for some neural networks

Conclusions

- Theoretical part:
  - NP-hardness
  - Explore submodularity for some neural networks
- Experimental part:
  - Practical method: gradient-guided greedy method
  - ★ We use sentence paraphrasing to expand the space of attacks
  - Experiments verified on three different tasks
  - Human Evaluation
  - ★ Adversarial training

Q & A

Thank you!

Methodology: Joint sentence and word paraphrasing attack

- Pick up sentence candidate set from semantic similarity.
- Greedily conduct sentence level paraphrasing attacks.

> I've always run jigdo-lite against my own mirror. It provides two things: 1) Proves ~~I can~~ you are able to build the ISOs from what I have mirrored locally. 2) Doesn't waste additional bandwidth. $\cdots$

- Pick up word candidate set from semantic and syntactic similarity.
- Greedily conduct word level paraphrasing attacks

> I've always run jigdo-lite against my own mirror. It ~~provides~~ offers two things: 1) Proves ~~I can~~ you are able to build the ISOs from what I have mirrored locally. 2) Doesn't waste additional bandwidth. As long as the checksums match what is provided from the official ISO image masters site, I don't see what the difference would be. Anyone else do this? ~~:)~~ ^_^

Experiment: Joint sentence and word paraphrasing attack

Table: Experiments on Word-level CNN with 5% dropout. [1] allows 50% word replacement while we only allow 20% word paraphrasing and 20% sentence paraphrasing.

Accuracy	Origin	ADV (ours)	ADV [1]
News	93.1%	35.4%	71.0%
Trec07p	99.1%	48.6%	64.5%
Yelp	93.6%	23.1%	39.0%

[1] V. Kuleshov, S. Thakoor, T. Lau, and S. Ermon, "Adversarial examples for natural language classification problems."

Experiments: Adversarial Training

Table: Performance of adversarial training.

Dataset	News	Trec07p	Yelp
Test (before)	93.1%	99.1%	93.6%
Test (after)	93.8%	99.2%	94.9%
ADV (before)	35.4%	48.6%	23.1%
ADV (after)	40.0%	54.2%	44.4%