Paper Presentation

Adversarial training methods for semi-supervised text classification

Baodi Shan

Nov 12, 2019


尴尬又不失礼貌的微笑

Table of Contents

Introduction

Last Week

- Have admiration for him
-
-

Last Week

- Have admiration for him
- Confused
-

Last Week

- Have admiration for him
- Confused
- In Despair

Story About Adversarial training

Figure 1: Dog

Figure 2: What?

- Why?

Story About Adversarial training

Figure 3: Dog

Figure 4: Noise

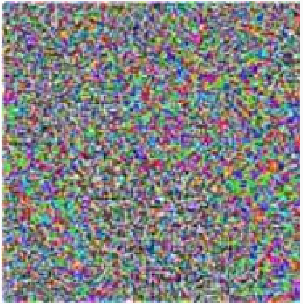
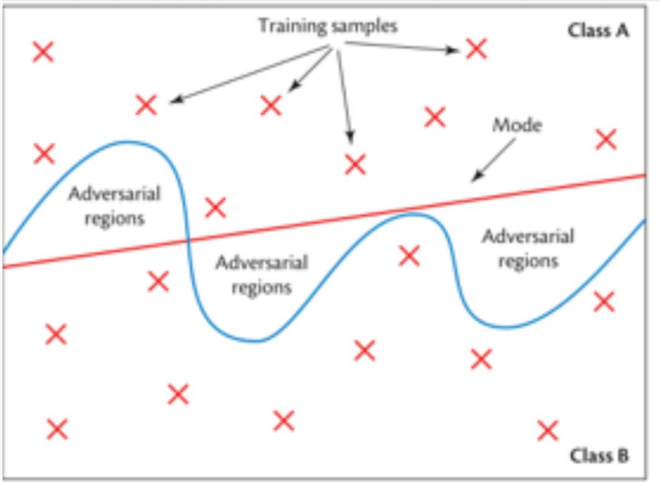


Figure 5: What?

- perturbations

NonLinear



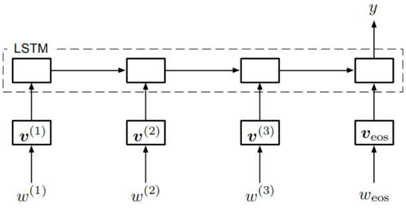
Training samples
Class A
Mode
Adversarial regions
Adversarial regions
Adversarial regions
Class B

Linear Behavior

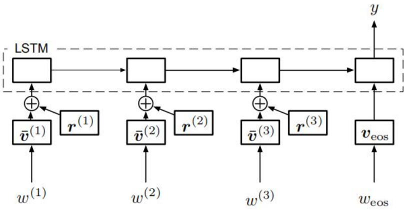
- The precision of an individual input feature is limited
- Example: 8 bits per pixel & 1/255 of the dynamic range
- for the classifier to respond differently to an input x than to an adversarial input $\widetilde{x} = x + \eta$ if every element of the perturbation η is smaller than the precision of the features

Models

Models



(a) LSTM-based text classification model.



(b) The model with perturbed embeddings.

Adversarial & Virtual Adversarial Trianing

Symbols

Table 1: explanation of symbols

symbol	explanation of symbols
x	input vector
I	input dimension
y	labels
Q	space of labels
$p(y\|x, \theta)$	input probability distributions
D_l	data with label
$D_u l$	data without label
$\dots$	$\dots$

Adversarial Training

- loss function:

$$L_{adv} := D[q(y|x_l), p(y|x_l + r_{adv}, \theta)]$$

$$where \quad r_{adv} := arg \quad max_{r:\|r\|\leqslant\epsilon} D[q(y|x_l), p(y|x_l + r, \theta)]$$

- solution:

$$r_{adv} \approx \epsilon \frac{g}{\|g\|^2}, where \quad g = \nabla_{x_l} D[h(y; y_l), p(y|x_l, \theta)]$$

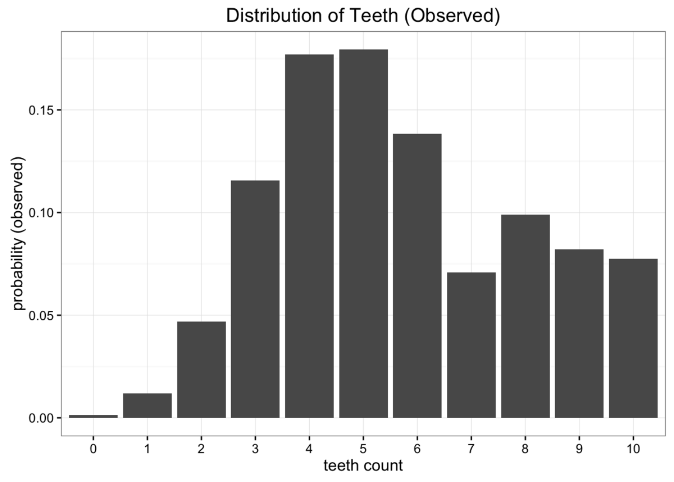
- or:

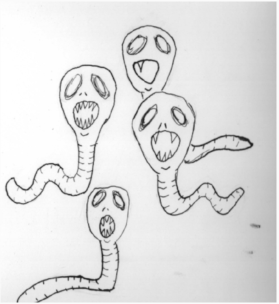
$$r_{adv} \approx \epsilon sign(g)$$

D? KL

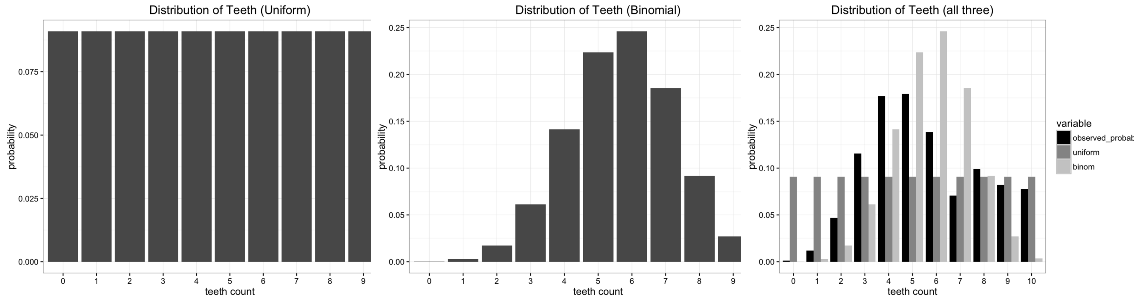
- Kullback–Leibler divergence
- How one probability distribution is different from a second.

Kullback–Leibler divergence





Kullback–Leibler divergence



Adversarial Training

- loss function:

$$L_{adv} := D[q(y|x_l), p(y|x_l + r_{adv}, \theta)]$$

$$where \quad r_{adv} := arg \quad max_{r:||r|| \leqslant \epsilon} D[q(y|x_l), p(y|x_l + r, \theta)]$$

- solution:

$$r_{adv} \approx \epsilon \frac{g}{||g||^2}, where \quad g = \nabla_{x_l} D[h(y; y_l), p(y|x_l, \theta)]$$

- or:

$$r_{adv} \approx \epsilon sign(g)$$

VAT

- target function:

$$L_{qadv} := D[q(y|x_*), p(y|x_* + r_{qadv}, \theta)]$$

$$where \quad r_{qadv} := arg \quad max_{r:\|r\| \leqslant \epsilon} D[q(y|x_*), p(y|x_* + r, \theta)]$$

- LDS and Regularization:

$$LDS(x_*, \theta) : D[q(y|x_*, \widehat{\theta}), p(y|x_* + r_{qadv}, \theta)]$$

$$where \quad r_{qadv} := arg \quad max_{r:\|r\| \leqslant \epsilon} D[q(y|x_*), p(y|x_* + r, \theta)]$$

loss function

$$R_{vadv}(D_l, D_{ul}, \theta) := \frac{1}{N_l + N_{ul}} \sum_{x_* \in D_L, D_{ul}} LDS(x_*, \theta)$$

$$loss = l(D_l, \theta) + \alpha R_{vadv}(D_l, D_{ul}, \theta)$$

Algorithm 1: Mini-batch SGD for $\nabla_\theta R_{vadv}(\theta)|_\theta = \widehat{\theta}$, with a one-time power iteration method

(1) Choose M samples of $x^{(i)}(i = 1, \cdots, M)$from dataset D at random.

(2) Generate a random unit vector $d^{(i)} \quad in \quad R^I$ using an iid Gaussian distribution.

(3) Calculate r_{vadv} via taking the gradient of D with respect to r on $r = \xi d^{(i)}$ on each input data point $x^{(i)}$:

$$g^{(i)} \leftarrow \nabla_r D[p(y|x^{(i)}, \widehat{\theta}), p(y|x^{(i)} + r, \widehat{\theta})]|_{r=\xi d^{(i)}}$$

$$r_{vadv}^{(i)} \leftarrow g^{(i)} / \|g^{(i)}\|_2$$

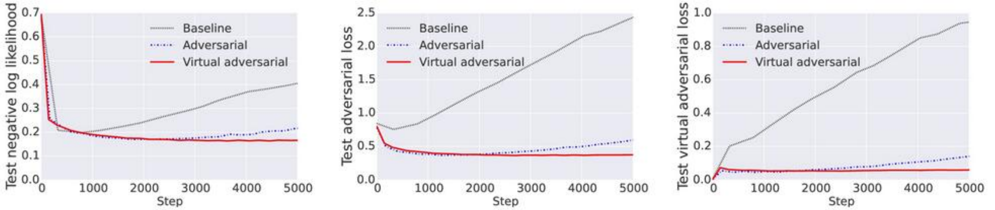
(4)Return $\nabla_\theta(\frac{1}{M}\sum_{i=1}^{M} D[p(y|x^{(i)}, \widehat{\theta}), p(y|x^{(i)} + r, \widehat{\theta})])$

Experiment and Conclusion

Dataset

	Classes	Train	Test	Unlabeled	Avg. T	Max T
IMDB	2	25,000	25,000	50,000	239	2,506
Elec	2	24,792	24,897	197,025	110	5,123
Rotten Tomatoes	2	9596	1066	7,911,684	20	54
DBpedia	14	560,000	70,000	–	49	953
RCV1	55	15,564	49,838	668,640	153	9,852

No Overfit



(a) Negative log likelihood (b) $L_{\mathrm{adv}}(\boldsymbol{\theta})$ (c) $L_{\text{v-adv}}(\boldsymbol{\theta})$

Result to IMDB

Method	Test error rate
Baseline (without embedding normalization)	7.33%
Baseline	7.39%
Random perturbation with labeled examples	7.20%
Random perturbation with labeled and unlabeled examples	6.78%
Adversarial	6.21%
Virtual Adversarial	**5.91%**
Adversarial + Virtual Adversarial	6.09%
Virtual Adversarial (on bidirectional LSTM)	**5.91%**
Adversarial + Virtual Adversarial (on bidirectional LSTM)	6.02%
Full+Unlabeled+BoW (Maas et al., 2011)	11.11%
Transductive SVM (Johnson & Zhang, 2015b)	9.99%
NBSVM-bigrams (Wang & Manning, 2012)	8.78%
Paragraph Vectors (Le & Mikolov, 2014)	7.42%
SA-LSTM (Dai & Le, 2015)	7.24%
One-hot bi-LSTM* (Johnson & Zhang, 2016b)	5.94%

Good or Bad?

	'good'				*'bad'*			
	Baseline	**Random**	**Adversarial**	**Virtual Adversarial**	**Baseline**	**Random**	**Adversarial**	**Virtual Adversarial**
1	great	great	decent	decent	terrible	terrible	terrible	terrible
2	decent	decent	great	great	awful	awful	awful	awful
3	×<u>bad</u>	excellent	nice	nice	horrible	horrible	horrible	horrible
4	excellent	nice	fine	fine	×<u>good</u>	×<u>good</u>	poor	poor
5	Good	Good	entertaining	entertaining	Bad	poor	BAD	BAD
6	fine	×<u>bad</u>	interesting	interesting	BAD	BAD	stupid	stupid
7	nice	fine	Good	Good	poor	Bad	Bad	Bad
8	interesting	interesting	excellent	cool	stupid	stupid	laughable	laughable
9	solid	entertaining	solid	enjoyable	Horrible	Horrible	lame	lame
10	entertaining	solid	cool	excellent	horrendous	horrendous	Horrible	Horrible

Results on Elec and RCV1

Method	Test error rate	
	Elec	RCV1
Baseline	6.24%	7.40%
Adversarial	5.61%	7.12%
Virtual Adversarial	5.54%	7.05%
Adversarial + Virtual Adversarial	**5.40%**	6.97%
Virtual Adversarial (on bidirectional LSTM)	5.55%	6.71%
Adversarial + Virtual Adversarial (on bidirectional LSTM)	5.45%	**6.68%**
Transductive SVM (Johnson & Zhang, 2015b)	16.41%	10.77%
NBLM (Naıve Bayes logisitic regression model) (Johnson & Zhang, 2015a)	8.11%	13.97%
One-hot CNN* (Johnson & Zhang, 2015b)	6.27%	7.71%
One-hot CNN† (Johnson & Zhang, 2016b)	5.87%	7.15%
One-hot bi-LSTM† (Johnson & Zhang, 2016b)	5.55%	8.52%

- $*$ indicates using pretrained embeddings of CNN, and † indicates using pretrained embeddings of CNN and bidirectional LSTM

Thanks to LaTeX and mtheme

Get the source of this theme and the demo presentation from

`github.com/matze/mtheme`

The theme *itself* is licensed under a Creative Commons Attribution-ShareAlike 4.0 International License.

OpenSource

Get the source code of this slide from

`github.com/lwshanbd/bigdata_slide`

Questions?

Thanks.