ADVERSARIAL TRAINING METHODS FOR SEMI-SUPERVISED TEXT CLASSIFICATION

Debabrata Kumar Karan
M.Tech AI (16858)
IISc, Bangalore
debabratak@iisc.ac.in

Abstract

Adversarial training provides a means of regularizing supervised learning algorithms while virtual adversarial training is able to extend supervised learning algorithms to the semi-supervised setting. However, both methods require making small perturbations to numerous entries of the input vector, which is inappropriate for sparse high-dimensional inputs such as one-hot word representations. adversarial and virtual adversarial training can be extended to the text domain by applying perturbations to the word embedding in a recurrent neural network rather than to the original input itself.

1 Introduction

Adversarial examples are examples that are created by making small perturbations to the input designed to significantly increase the loss incurred by a machine learning model. Several models, including state of the art convolutional neural networks, lack the ability to classify adversarial examples correctly, sometimes even when the adversarial perturbation is constrained to be so small that a human observer cannot perceive it. Adversarial training is the process of training a model to correctly classify both unmodified examples and adversarial examples. It improves not only robustness to adversarial examples, but also generalization performance for original examples. Adversarial training requires the use of labels when training models that use a supervised cost, because the label appears in the cost function that the adversarial perturbation is designed to maximize. Virtual adversarial training extends the idea of adversarial training to the semi-supervised regime and unlabeled examples. This is done by regularizing the model so that given an example, the model will produce the same output distribution as it produces on an adversarial perturbation of that example. Virtual adversarial training achieves good generalization performance for both supervised and semi-supervised learning tasks.

Adversarial perturbations typically consist of making small modifications to very many real-valued inputs. For text classification, the input is discrete, and usually represented as a series of high dimensional one-hot vectors. Because the set of high-dimensional one-hot vectors does not admit infinitesimal perturbation, I define the perturbation on continuous word embedding instead of discrete word inputs. Traditional adversarial and virtual adversarial training can be interpreted both as a regularization strategy and as defense against an adversary who can supply malicious inputs. Since the perturbed embedding does not map to any word and the adversary presumably does not have access to the word embedding layer, This training strategy is no longer intended as a defense against an adversary. so propose this approach exclusively as a means of regularizing a text classifier by stabilizing the classification function.
This approach with neural language model unsupervised pretraining achieves state of the art performance for multiple semi-supervised text classification tasks, including sentiment classification and topic classification. .

2 Related Work

Dropout is a regularization method widely used for many domains including text. There are some previous works adding random noise to the input and hidden layer during training, to prevent over fitting. However, in my experiments training with adversarial and virtual adversarial perturbations outperformed the method with random perturbations. For semi-supervised learning, a common approach, especially in the image domain, is to train a generative model whose latent features may be used as features for classification .

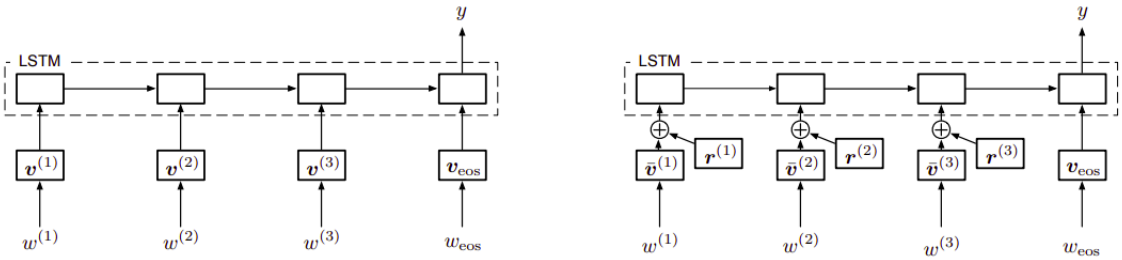
These models now achieve state of the art performance on the image domain. However, these methods require numerous additional hyper parameters with generative models, and the conditions under which the generative model will provide good supervised learning performance are poorly understood. By comparison, adversarial and virtual adversarial training requires only one hyper parameter, and has a straightforward interpretation as robust optimization. There has also been semi-supervised approaches applied to text classification with both CNNs and RNNs. These approaches utilize 'view-embeddings' which use the window around a word to generate its embedding. When these are used as a pretrained model for the classification model, they are found to improve generalization performance.

3 Proposed Approach

A simple bidirectional LSTM-based neural network model with ADAM optimiser and customized binary cross entropy loss function is used.

3.1 Model

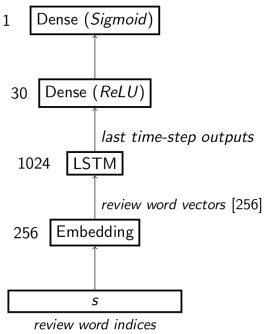
let, word $\{w^t | t = 1, 2, ..., T\}$ and target y. To transform a discrete word input to a continuous vector let $V \in \mathbb{R}^{(K+1)*D}$ where K is the number of words in the vocabulary and each row v_k corresponds to the word embedding of the i^{th} word. As a text classification model, here a simple LSTM-based neural network model is used , At time step t, the input is the discrete word w^k, and the corresponding word embedding is v^t.



(a) LSTM-based text classification model.

(b) The model with perturbed embeddings.

For constructing the bidirectional LSTM model for text classification, I add an additional LSTM on the reversed sequence to the unidirectional LSTM model described in fig. The model then predicts the label on the concatenated LSTM outputs of both ends of the sequence.



In adversarial and virtual adversarial training, we train the classifier to be robust to perturbations of the embeddings.The perturbations are of bounded norm. The model could trivially learn to make the perturbations insignificant by learning embeddings with very large norm. To prevent this pathological solution, apply adversarial and virtual adversarial training to the model as defined above, and replace the embeddings v_k with normalized embeddings $\bar{v_k}$, defined as:

$\bar{v_k} = \frac{v_k - E(v)}{\sqrt{var(v)}}$, where $E(v) = \sum_{j=1}^{K} f_j v_j$, $var(v) = \sum_{j=1}^{K} f_j (v_j - E(v))^2$

where f_i is the frequency of the i^{th} word, calculated within all training examples.

3.2 Loss Function

3.2.1 Adversarial training

Adversarial training is a novel regularization method for classifiers to improve robustness to small, approximately worst case perturbations.

Let us denote x as the input and θ as the parameters of a classifier. When applied to a classifier, adversarial training adds the following term to the cost function:

$-\log p(y|x + r_{adv}; \theta)$, where $r_{adv} = \underset{r, \|r\| < \epsilon}{\arg\min} \log p(y|x + r; \hat{\theta})$

where r is a perturbation on the input and $\hat{\theta}$ is a constant set to the current parameters of a classifier. The use of the constant copy $\hat{\theta}$ rather than θ indicates that the back propagation algorithm should not be used to propagate gradients through the adversarial example construction process. At each step of training, identify the worst case perturbations r_{adv} against the current model $p(y|x; \hat{\theta})$ in above Eq., and train the model to be robust to such perturbations through minimizing above Eq. with respect to θ. However, we cannot calculate this value exactly in general, because exact minimization with respect to r is intractable for many interesting models such as neural networks. By linearizing $\log p(y|x; \hat{\theta})$ around x we can solve this problem. With a linear approximation and a L2 norm constraint in above Eq., the resulting adversarial perturbation is

$r_{adv} = -\epsilon \frac{g}{\|g\|_2}$, where $g = \nabla_x \log p(y|x; \hat{\theta})$.

To be robust to the adversarial perturbation defined in above Eq.,the adversarial loss:

$L_{adv} = -\frac{1}{N} \sum_{n=1}^{N} \log p(y_n|x_n + r_{adv,n}; \theta)$, where N is the number of labeled examples.

3.2.2 Virtual adversarial training

Virtual adversarial training is a regularization method closely related to adversarial training. The additional cost introduced by virtual adversarial training is the following:

$KL[p(.|x; \hat{\theta}) \| p(.|x + r_{v\_adv}; \theta)]$ where $r_{v\_adv} = \underset{r, \|r\| < \epsilon}{\arg\max} KL[p(.|x; \hat{\theta}) \| p(.|x + r; \hat{\theta})]$

where $KL[p||q]$ denotes the KL divergence between distributions p and q. By minimizing above Eq., a classifier is trained to be smooth. This can be considered as making the classifier resistant to perturbations in directions to which it is most sensitive on the current model $p(y|x; \hat{\theta})$. Virtual adversarial loss in above Eq. requires only the input x and does not require the actual label y like in adversarial loss which requires the label y. This makes it possible to apply virtual adversarial training to semi-supervised learning.

The virtual adversarial loss is defined as:

$L_{v\_adv} = \frac{1}{N'} \sum_{n'=1}^{N'} KL[p(.|x_{n'}; \hat{\theta}) \| p(.|x_{n'} + r_{v\_adv,n'}; \theta)]$, where N' is the number of both labeled and unlabeled examples.

4 Experiment

All experiments used TensorFlow on GPUs. Code will be available at github

`https://github.com/debabratakaran/DLNLP-term-project.git`

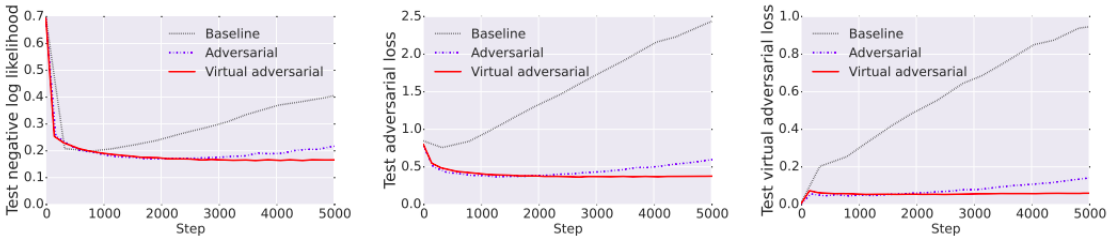
4.1 Dataset

	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

4.2 Result

I tested on IMDB dataset

Negative log likelihood curve of three methods



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

Accuracy on different model with best sequence length and 10 epochs

Method	Seq. Length	Epochs	Accuracy
baseline	400	10	0.906
adversarial	400	10	0.914
virtual adv.	400	10	**0.921**
baseline	600	10	0.904
adversarial	600	10	0.912
pyramidal baseline	1200	10	0.910
pyramidal adversarial	1200	10	0.916

5 References

[1] Takeru Miyato & Andrew M Dai & Ian Goodfellow , *Adversarial training methods for semi-supervised text classification*, pp. Preferred Networks, Inc., ATR Cognitive Mechanisms Laboratories, Kyoto University Google Brain OpenAI