Revisiting LSTM Networks for Semi-Supervised Text Classification via Mixed Objective Function

Devendra Singh Sachan
Petuum, Inc
Pittsburgh, PA, USA
sachan.devendra@gmail.com

Manzil Zaheer
Machine Learning Department, CMU
Pittsburgh, PA, USA
manzilz@andrew.cmu.edu

Ruslan Salakhutdinov
Machine Learning Department, CMU
Pittsburgh, PA, USA
rsalakhu@andrew.cmu.edu

ABSTRACT

In this paper, we do a careful study of a bidirectional LSTM network for the task of text classification using both supervised and semi-supervised approaches. In prior work, it has been reported that in order to get good classification accuracy using LSTM models for text classification task, pretraining the LSTM model parameters using unsupervised learning methods such as language modeling or sequence auto-encoder is necessary [2, 20]. However, we find that our simple model, when trained with cross-entropy loss is able to achieve competitive results compared with the more complex models. Furthermore, in addition to cross-entropy loss, by using a combination of entropy minimization, adversarial, and virtual adversarial losses for both labeled and unlabeled data, we report new state-of-the-art results for text classification task on four benchmark datasets. In particular, on ACL-IMDB sentiment analysis and AG-News topic classification datasets, our method outperforms current approaches by a substantial margin.

CCS CONCEPTS

• **Computing methodologies** → **Neural networks**; **Learning latent representations**; *Maximum entropy modeling*;

KEYWORDS

Text Classification, Semi-Supervised Learning, Adversarial Training, LSTM

1 INTRODUCTION

Text classification is an important problem in natural language processing (NLP) where the task is to assign a document to one or more predefined categories. It has a wide range of applications such as sentiment analysis [23], topic categorization [14], and email filtering. Earlier machine learning approaches to this task were based on the extraction of bag-of-words [5] features such as tf-idf weighting, followed by a supervised classifier such as Naive Bayes [17] or a linear SVM [7]. Although simple and effective, these approaches ignore the word order structure present in the document, suffer from data sparsity and high-dimensionality problems.

Recently, neural network approaches for NLP are getting increasingly popular as these models can be trained end-to-end and have yielded impressive results on several benchmark tasks [1, 24]. Standard neural network approaches for text classification involve using recurrent neural networks like LSTMs [2, 26] or one-dimensional CNNs [12] on top of pretrained word vectors. Many variations of these models have been proposed in order to improve the classification accuracy for this task: Johnson and Zhang [8, 10] train a one-hot CNN and a one-hot bidirectional LSTM (BiLSTM) network respectively with dynamic max-pooling over time.

However, it has been reported in previous work that it is difficult to optimize the LSTM network for text classification tasks and therefore in order to perform well, its parameters required pretraining with either a language model or a sequence auto-encoder [2]. A disadvantage of this approach is that it can take a long time to train a language model or a sequence auto-encoder and thus this additional step may not be always feasible. In this paper, we show that such pretraining of LSTM model parameters is not necessary in order to train an LSTM network that achieves competitive results with respect to the more complex approaches.

We also explore the applicability of semi-supervised learning (SSL) techniques to text classification. Previous work regarding the application of SSL to this task includes training Naive Bayes model on unlabeled data using expectation-maximization algorithm [22], incorporation of multi-view embeddings representation of input words along with its one-hot representation as model's input [9], use of additional adversarial and virtual adversarial losses for model training [20]. We use one-layer BiLSTM model with max-pooling and a combination of cross-entropy, entropy minimization, adversarial, and virtual adversarial training to obtain a substantial improvement in classification accuracy on four benchmark datasets compared with the more complex methods. We also perform ablation studies on the ACL-IMDB dataset to know the factors that result in an improved performance.

2 METHODS

In this section, we will describe the model architecture, supervised loss functions, unsupervised loss functions, and the mixed objective function. For mathematical notation, we will use bold lowercase to denote vectors, bold uppercase to denote matrices, and lowercase to denote scalars and individual words in a document.

2.1 Model Architecture

The classification model consists of an embedding layer, a bidirectional long-short-term-memory (BiLSTM) encoder [6, 25], a max-pooling layer, a linear layer, and finally a softmax layer (see Figure 1). First, the sequence of words $(w_1, \ldots, w_T)$ contained in a document are passed through an embedding layer that contains a lookup table which maps them to dense vectors $(\mathbf{v}_1, \ldots, \mathbf{v}_T, \ \mathbf{v}_t \in \mathbb{R}^d)$.

KDD'18 Deep Learning Day, August 2018, London, UK

ACM ISBN 978-x-xxxx-xxxx-x/YY/MM.
https://doi.org/10.1145/nnnnnnn.nnnnnnn

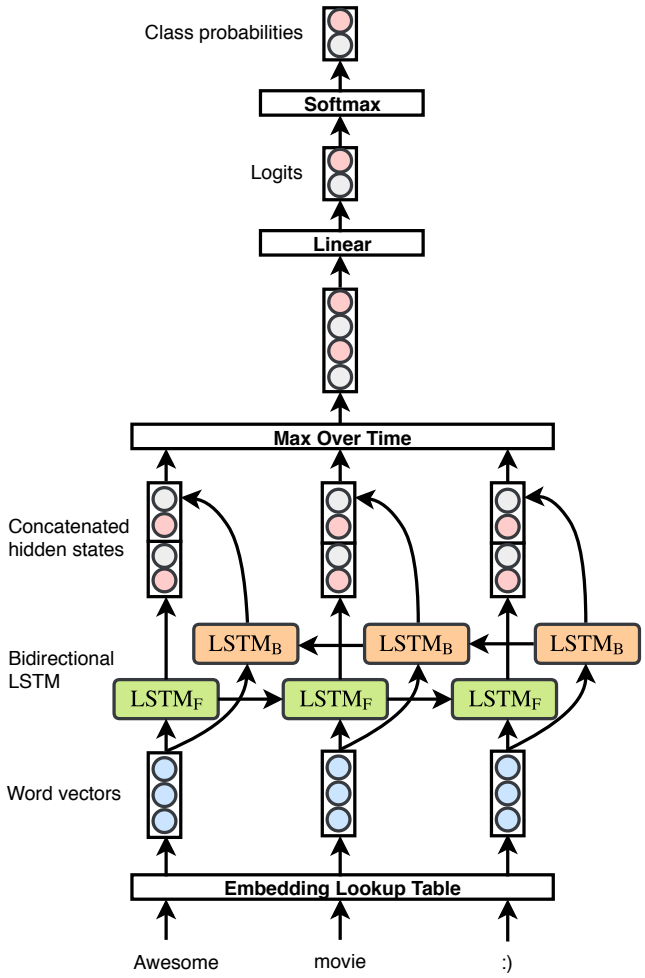

Figure 1: Text classification model architecture. The words are mapped to dense vectors via embedding layer lookup and are given as input to the BiLSTM. The hidden states from BiLSTM are concatenated and then fed to the max-pooling layer that outputs feature representations. These features are next fed to a classifier that computes class probabilities.

Next, the forward LSTM and backward LSTM of the BiLSTM encoder processes these word vectors in the forward (left to right) and backward (right to left) directions respectively and produces corresponding hidden states at each time-step

$$\overrightarrow{\mathbf{h}_t} = \overrightarrow{\text{LSTM}_\text{F}}(\mathbf{v}_t), \ \overleftarrow{\mathbf{h}_t} = \overleftarrow{\text{LSTM}_\text{B}}(\mathbf{v}_t).$$

Next, these hidden state outputs from the forward LSTM ($\overrightarrow{\mathbf{h}_t}$) and backward LSTM ($\overleftarrow{\mathbf{h}_t}$) are concatenated at every time-step to enable encoding of information from past and future contexts respectively

$$\mathbf{h}_t = [\overrightarrow{\mathbf{h}_t} \ || \ \overleftarrow{\mathbf{h}_t}], \ t \in [1, T], \ \mathbf{h}_t \in \mathbb{R}^n.$$

These concatenated hidden states are next fed to the pooling layer that computes the maximum value over time to obtain the feature representation of the input sequence ($\mathbf{h} \in \mathbb{R}^n$)

$$h^\ell = \max_t h_t^\ell, \ t \in [1, T], \ \forall \ell \in \{1, \ldots, n\}.$$

This max-pooling mechanism constrains the model to capture the most useful features produced by the BiLSTM encoder. Next, the linear layer applies an affine transformation to the feature vector to produce *logits* ($\mathbf{d}$)

$$\mathbf{d} = \mathbf{W}\mathbf{h} + b, \ \mathbf{d} \in \mathbb{R}^K,$$

where K is the number of classes, $\mathbf{W}$ is the weight matrix, and b is the bias. Next, these logits are normalized using the softmax function to give our estimated class probabilities as

$$p(y = k \mid \mathbf{x}; \boldsymbol{\theta}) = \frac{\exp(d_k)}{\sum_{j=1}^{K} \exp(d_j)}, \ \forall k \in \{1, \ldots, K\},$$

where $(\mathbf{x}, y)$ is a training example and $\boldsymbol{\theta}$ denotes the model parameters.

2.2 Supervised Training

Let there be m_ℓ labeled examples in the training set that are denoted as $\{(\mathbf{x}^{(1)}, y^{(1)}), \ldots, (\mathbf{x}^{(m_\ell)}, y^{(m_\ell)})\}$, where $\mathbf{x}^{(i)}$ represents a document's word sequence, $y^{(i)}$ represents class label such that $y^{(i)} \in \{1, 2, \ldots, K\}$. For supervised training of the classification model, we make use of two methodologies: *maximum likelihood* estimation and *adversarial training*, which are described next.

Maximum Likelihood (ML). The most widely used method to learn the parameters of neural network models for text classification task is to perform maximum likelihood training of the observed data. This is achieved if we minimize the average cross-entropy loss between the estimated class probability and the ground truth class label for all training examples

$$L_{ML}(\boldsymbol{\theta}) = -\frac{1}{m_\ell} \sum_{i=1}^{m_\ell} \sum_{k=1}^{K} \mathbb{1}(y^{(i)} = k) \log p(y^{(i)} = k \mid \mathbf{x}^{(i)}; \boldsymbol{\theta}),$$

where $\mathbb{1}(;)$ is an indicator function.

Adversarial Training (AT). Adversarial examples are created from inputs by inducing small perturbations in them to mislead the machine learning algorithm. The objective of adversarial training is to construct and give as input adversarial examples during model training procedure to make the model more robust to adversarial noise and thereby improving its generalization ability [3].

In this work, we make adversarial perturbations to the input word embeddings ($\mathbf{v} = [\mathbf{v}_1, \ldots, \mathbf{v}_T]$) [20]. These perturbations ($\mathbf{r}_{adv}$) are estimated by linearizing the supervised cross-entropy loss around the input word embeddings. Specifically, to get the adversarial embedding ($\mathbf{v}^*$) corresponding to $\mathbf{v}$, we use the L_2 norm of the training loss gradient ($\mathbf{g}$) that is computed by backpropagation using the current model parameters ($\hat{\boldsymbol{\theta}}$).

$$\mathbf{r}_{adv} = \epsilon \mathbf{g} / \|\mathbf{g}\|_2, \ \text{where } \mathbf{g} = -\nabla_\mathbf{v} \log p(y = k \mid \mathbf{v}; \hat{\boldsymbol{\theta}})$$
$$\mathbf{v}^* = \mathbf{v} + \mathbf{r}_{adv}$$

where, k is the correct class label, ϵ is a hyperparameter that controls the magnitude of the perturbation. We apply adversarial loss to

only the labeled data. It is defined as

$$L_{AT}(\boldsymbol{\theta}) = -\frac{1}{m_\ell}\sum_{i=1}^{m_\ell}\sum_{k=1}^{K}\mathbb{1}(y^{(i)}=k)\log p(y^{(i)}=k \mid \mathbf{v}^{*(i)};\hat{\boldsymbol{\theta}}).$$

2.3 Unsupervised Training

Let there be an additional m_u unlabeled examples in the dataset that are denoted as $\{\mathbf{x}^{(1)},\ldots,\mathbf{x}^{(m_u)}\}$. In this paper, in addition to supervised training, we also experiment with two unsupervised methodologies: *entropy minimization* and *virtual adversarial training*. These loss terms when incorporated into the objective function act as effective regularizers during model training. Next, we describe them in detail.

2.3.1 ***Entropy Minimization (EM).*** In addition to supervised cross-entropy loss, we also minimize the conditional entropy of the estimated class probabilities [4, 21]. This can also be interpreted as a special case of the missing label problem where the probability $p(y^{(i)}=k \mid \mathbf{x}^{(i)};\boldsymbol{\theta})$ signifies a soft assignment of the i^{th} example to label k (*i.e.* soft clustering). Entropy minimization loss is applied in an unsupervised manner to both the labeled and unlabeled data.

$$L_{EM}(\boldsymbol{\theta}) = -\frac{1}{m}\sum_{i=1}^{m}\sum_{k=1}^{K}p(y^{(i)}=k \mid \mathbf{x}^{(i)};\boldsymbol{\theta})\log p(y^{(i)}=k \mid \mathbf{x}^{(i)};\boldsymbol{\theta}),$$

where $m = m_\ell + m_u$.

2.3.2 ***Virtual Adversarial Training (VAT).*** As opposed to minimizing the cross-entropy loss of the adversarial examples in AT, in VAT, we minimize the KL divergence between $p(\mathbf{v})$ and $p(\mathbf{v}^*)$, where $\mathbf{v}^* = \mathbf{v} + \mathbf{r}_{vadv}$. The motivation of using KL divergence as an additional loss term in the objective function is that it tends to make the loss surface smooth at the current example [21]. Also, computing the VAT loss doesn't require class labels, so it can be applied to unlabeled data as well. In this work, we follow the approach proposed by Miyato et al. [20] that makes use of the second-order Taylor expansion of distance followed by power iteration method to approximate the virtual adversarial perturbation. First, an i.i.d. random unit vector is sampled for every example from the Normal distribution ($\mathbf{d}^{(i)} \sim \mathcal{N}(\mathbf{0},\mathbf{I}) \in \mathbb{R}^d$) and then adversarial perturbation computed as $\xi\mathbf{d}^{(i)}$ is added to the word embeddings, where ξ is a hyperparameter

$$\mathbf{v}'^{(i)} = \mathbf{v}^{(i)} + \xi\mathbf{d}^{(i)}.$$

Next, the gradient is estimated from the KL divergence as shown below

$$\mathbf{g} = \nabla_{\mathbf{v}'} D_{KL}(p(. \mid \mathbf{v}^{(i)};\hat{\boldsymbol{\theta}}) \parallel p(. \mid \mathbf{v}'^{(i)};\hat{\boldsymbol{\theta}})).$$

Next, virtual adversarial perturbation ($\mathbf{r}_{vadv}$) is generated using the L_2 norm of the gradient and added to the word embeddings

$$\mathbf{r}^{(i)}_{vadv} = \epsilon\mathbf{g}/\|\mathbf{g}\|_2$$
$$\mathbf{v}^{*(i)} = \mathbf{v}^{(i)} + \mathbf{r}^{(i)}_{vadv}.$$

Lastly, virtual adversarial loss can be computed from both the labeled and unlabeled data as:

$$L_{VAT}(\boldsymbol{\theta}) = \frac{1}{m}\sum_{i=1}^{m} D_{KL}(p(. \mid \mathbf{v}^{(i)};\boldsymbol{\theta}) \parallel p(. \mid \mathbf{v}^{*(i)};\boldsymbol{\theta})),$$

where $m = m_\ell + m_u$.

Dataset	Training	Test	K	ℓ
ACL-IMDB	25K	25k	2	268
Elec	25K	25K	2	125
AG-News	120K	7.6K	4	46
DBpedia	560K	70K	14	56

Table 1: Dataset summary statistics. K: Number of classes. ℓ: Average length of a document in the training set.

Dataset	Batchsize	Vocab	ϵ
ACL-IMDB	3000	80K	5
Elec	2000	40K	2
AG-News	2000	75K	1
DBpedia	7500	50K	1

Table 2: Hyperparameter settings for various datasets. Batchsize refers to the number of tokens in a minibatch, ϵ refers to the adversarial perturbation.

2.4 Mixed Objective Function

Our mixed objective function combines the above described supervised and unsupervised loss functions using λ_{ML}, λ_{AT}, λ_{EM}, and λ_{VAT} as hyperparameters

$$L_{MIXED}(\boldsymbol{\theta}) = \lambda_{ML}L_{ML}(\boldsymbol{\theta}) + \lambda_{AT}L_{AT}(\boldsymbol{\theta}) + \lambda_{EM}L_{EM}(\boldsymbol{\theta}) + \lambda_{VAT}L_{VAT}(\boldsymbol{\theta}).$$

3 EXPERIMENTS

Dataset Description

In this work, we experiment with four datasets that are summarized in Table 1. ACL-IMDB [15] and Elec [8] datasets are for sentiment classification of movie reviews and Amazon product reviews [16] respectively while AG-News [27] and DBpedia [27] are for topic classification of news articles and Wikipedia articles respectively. We preprocess the datasets by converting all the text to lowercase.

Experimental Setup

Our setup is common for all experiments. We initialize the parameters of the embedding layer with pretrained word embeddings. We learn embeddings for the ACL-IMDB dataset using `word2vec` [19] and use fastText pretrained embeddings [18] for all the other datasets. We use 512-D BiLSTM hidden states and apply dropout (p_{drop} = 0.5) on the word embeddings and on the LSTM hidden states. Adam optimizer [13] is used with *learning rate* = $1e^{-3}$, $\beta_1 = 0$, $\beta_2 = 0.98$, $\epsilon_{adam} = 1e^{-8}$ and we follow exponential learning rate decay. We perform gradient clipping using a maximum norm of 1.0. For semi-supervised experiments, we additionally include all the losses described above with λ_{ML}, λ_{AT}, λ_{EM}, λ_{VAT} set to 1.0 and $\xi = 0.1$. Also, in contrast to Miyato et al. [20], we don't perform any embedding layer normalization during AT or VAT training. Dataset-specific hyperparameters are mentioned in Table 2. We did not perform any sequence truncation for training. Therefore, to prevent out of memory issues because of longer sequences, we use a dynamic batch size

Method	ACL-IMDB	Elec	AG-News	DBpedia
Johnson and Zhang [9, 11]	7.67	**7.14**	6.88	**0.88**
Zhang et al. [27]	-	-	9.51	1.55
Our Method (L_{ML})	**6.40**	7.40	**5.62**	0.91

Table 3: Test set error rates when the model is trained using maximum likelihood objective (L_{ML}).

Method	ACL-IMDB	Elec	AG-News	DBpedia
Johnson and Zhang [9, 11]	7.67	**7.14**	6.88	**0.88**
Johnson and Zhang [10, 11]	5.94	5.55	6.57	0.84
Miyato et al. [20]	5.91	5.40	-	0.76
Dai and Le [2]	7.24	-	-	1.19
Our Method (L_{MIXED})	**4.32**	**5.24**	**4.95**	**0.70**

Table 4: Test set error rates when the model is trained using mixed objective function (L_{MIXED}).

Method	ACL-IMDB
Baseline	93.43
+Word Dropout	**93.57**
Random Embeddings	92.36
Static Embedding	91.83
5K words per batch	**93.50**
1K words per batch	93.02
256-D hidden size	93.33
64 batch size	93.14
32 batch size	**93.60**
Default Adam parameters	93.37
30K Vocab	92.22
300 max length	93.31
gradient norm (5.0)	92.99

Table 5: Test accuracies on ACL-IMDB dataset for variants of our best BiLSTM model trained using maximum likelihood objective (L_{ML}).

of fixed words per minibatch. For supervised experiments (L_{ML}), we perform training till 10 epochs and for semi-supervised experiments (L_{MIXED}), training is done till 50 epochs. For ACL-IMDB and Elec datasets, we use training and test set as the unlabeled data, while for AG-News and DBpedia datasets, we use only the training set as the unlabeled data. The batch size of the unlabeled data is kept the same as that of the labeled data.

4 RESULTS

In this section, we report the classification accuracy on the test set and do ablation studies when training the model using both supervised and unsupervised loss functions.

We show our results and related baseline scores when using only the maximum likelihood objective in Table 3. We observe that using only single-layer BiLSTM encoder having pretrained word embeddings when trained with cross-entropy loss achieves very competitive performance for all datasets as compared to the more complex approaches such as one-hot LSTM or pyramidal CNN [10, 11]. For ACL-IMDB and AG-News datasets, we report better results when considering only supervised cross-entropy loss during model training.

We also want to emphasize that unlike Dai and Le [2] or Miyato et al. [20], our approach doesn't require any pretraining of the BiLSTM weights. To know the importance of various components of the model and also our training strategy, we perform ablation studies on the ACL-IMDB dataset whose results are shown in Table 5. We note that good performance of our model mostly comes from finetuning the pretrained embedding layer, using a larger vocabulary size, and using a lower threshold value for gradient norm clipping. Including input word dropout and increasing the batch size also helps to improve performance. Surprisingly, we also note that using a smaller LSTM network, sequence length truncation, and having default Adam optimizer parameters results in only a small drop in test accuracy.

In our next set of experiments, we include adversarial training, entropy minimization, and virtual adversarial training to the maximum likelihood objective (L_{MIXED}). From the results shown in Table 4, we observe that with the addition of these loss functions, we get new state-of-the-art results on all the four datasets. Specifically, on ACL-IMDB, there is a substantial reduction of 26.9% in relative error rate compared with the approach of Johnson and Zhang [10], in which they use one-hot LSTM along with a variety of multi-view embeddings as additional features. Although similar in formulation, our approach also performs much better than Miyato et al. [20] where they also use language modeling pretraining. Similarly, for AG-News dataset, we observe an error reduction of 26.6% relative to Johnson and Zhang [11] who apply a deep pyramidal CNN along with unsupervised embeddings as additional features. Even on the other two datasets, the results are an improvement over the previous best semi-supervised results. We also note that, with the use of these additional loss functions, the accuracy on ACL-IMDB dataset has drastically improved by 32.5% and on Elec dataset has

Method	ACL-IMDB
ML Supervised Baseline	93.43
+ VAT (L)	94.25
+ VAT (L + U)	94.68
+ VAT (L + U) + EM	**95.53**
+ VAT (L + U) + Word Dropout	95.26
+ VAT (L + 4×U) + EM	94.63

Table 6: Test accuracy on ACL-IMDB dataset when trained using mixed objective function (L_{MIXED}) on variants of our best BiLSTM model. L: labeled data. U: unlabeled data, 4×U implies that the size of unlabeled data was four times to that of labeled data.

improved by 30.25% compared with using only the cross-entropy loss. Similarly, for AG-News and DBpedia datasets, we observe a significant reduction in the test error rates.

Next, we analyze the effect of various unsupervised loss functions on ACL-IMDB dataset in Table 6. We observe that the inclusion of VAT loss leads to significant performance gains. Furthermore, when the VAT loss is combined with entropy minimization loss, we see 18.8% increase in accuracy. A surprising result is that the addition of word dropout leads to a decrease in performance. Also, training using unlabeled data of much greater size than supervised data results in a performance degradation.

5 CONCLUSION

We show that a single-layer BiLSTM classifier using maximum likelihood training results in a competitive or better performance on a variety of text classification tasks without the need of language model pretraining. Also, in addition to maximum likelihood objective, using a combination of entropy minimization loss, adversarial, and virtual adversarial training, we report new state-of-the-art results on four benchmark datasets for text classification.

REFERENCES

[1] Ronan Collobert, Jason Weston, Léon Bottou, Michael Karlen, Koray Kavukcuoglu, and Pavel Kuksa. 2011. Natural Language Processing (Almost) from Scratch. *J. Mach. Learn. Res.* 12 (Nov. 2011), 2493–2537.

[2] Andrew M. Dai and Quoc V. Le. 2015. Semi-supervised Sequence Learning. In *Proceedings of the 28th International Conference on Neural Information Processing Systems - Volume 2 (NIPS'15)*. MIT Press, Cambridge, MA, USA, 3079–3087.

[3] Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. 2014. Explaining and harnessing adversarial examples. *Computing Research Repository* arXiv:1412.6572 (2014).

[4] Yves Grandvalet and Yoshua Bengio. 2004. Semi-supervised Learning by Entropy Minimization. In *Proceedings of the 17th International Conference on Neural Information Processing Systems (NIPS'04)*. MIT Press, Cambridge, MA, USA, 529–536.

[5] Zellig S Harris. 1954. Distributional structure. *Word* 10, 2-3 (1954), 146–162.

[6] Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long Short-Term Memory. *Neural Comput.* 9, 8 (Nov. 1997), 1735–1780.

[7] Thorsten Joachims. 1998. Text Categorization with Support Vector Machines: Learning with Many Relevant Features. In *Proceedings of the 10th European Conference on Machine Learning (ECML'98)*. Springer-Verlag, Berlin, Heidelberg, 137–142.

[8] Rie Johnson and Tong Zhang. 2015. Effective Use of Word Order for Text Categorization with Convolutional Neural Networks. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. Association for Computational Linguistics, Denver, Colorado, 103–112.

[9] Rie Johnson and Tong Zhang. 2015. Semi-supervised Convolutional Neural Networks for Text Categorization via Region Embedding. In *Proceedings of the 28th International Conference on Neural Information Processing Systems - Volume 1 (NIPS'15)*. MIT Press, Cambridge, MA, USA, 919–927.

[10] Rie Johnson and Tong Zhang. 2016. Supervised and Semi-supervised Text Categorization Using LSTM for Region Embeddings. In *Proceedings of the 33rd International Conference on International Conference on Machine Learning - Volume 48 (ICML'16)*. JMLR.org, 526–534.

[11] Rie Johnson and Tong Zhang. 2017. Deep Pyramid Convolutional Neural Networks for Text Categorization. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Association for Computational Linguistics, Vancouver, Canada, 562–570.

[12] Yoon Kim. 2014. Convolutional Neural Networks for Sentence Classification. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Association for Computational Linguistics, Doha, Qatar, 1746–1751.

[13] Diederik P. Kingma and Jimmy Ba. 2014. Adam: A Method for Stochastic Optimization. *Computing Research Repository* arXiv:1412.6980 (2014).

[14] David D. Lewis, Yiming Yang, Tony G. Rose, and Fan Li. 2004. RCV1: A New Benchmark Collection for Text Categorization Research. *J. Mach. Learn. Res.* 5 (Dec. 2004), 361–397.

[15] Andrew L. Maas, Raymond E. Daly, Peter T. Pham, Dan Huang, Andrew Y. Ng, and Christopher Potts. 2011. Learning Word Vectors for Sentiment Analysis. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*. Association for Computational Linguistics, Portland, Oregon, USA, 142–150.

[16] Julian McAuley and Jure Leskovec. 2013. Hidden Factors and Hidden Topics: Understanding Rating Dimensions with Review Text. In *Proceedings of the 7th ACM Conference on Recommender Systems (RecSys '13)*. ACM, New York, NY, USA, 165–172.

[17] Andrew McCallum and Kamal Nigam. 1998. A comparison of event models for Naive Bayes text classification. In *Proceedings of AAAI-98, Workshop on Learning for Text Categorization*. AAAI Press, 41–48.

[18] Tomas Mikolov, Edouard Grave, Piotr Bojanowski, Christian Puhrsch, and Armand Joulin. 2018. Advances in Pre-Training Distributed Word Representations. In *Proceedings of the International Conference on Language Resources and Evaluation (LREC 2018)*.

[19] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Distributed Representations of Words and Phrases and Their Compositionality. In *Proceedings of the 26th International Conference on Neural Information Processing Systems - Volume 2 (NIPS'13)*. Curran Associates Inc., USA, 3111–3119.

[20] Takeru Miyato, Andrew M Dai, and Ian Goodfellow. 2016. Adversarial training methods for semi-supervised text classification. *Computing Research Repository* arXiv:1605.07725 (2016).

[21] Takeru Miyato, Shin-ichi Maeda, Masanori Koyama, and Shin Ishii. 2017. Virtual adversarial training: a regularization method for supervised and semi-supervised learning. *Computing Research Repository* arXiv:1704.03976 (2017).

[22] Kamal Nigam, Andrew Kachites McCallum, Sebastian Thrun, and Tom Mitchell. 2000. Text Classification from Labeled and Unlabeled Documents Using EM. *Mach. Learn.* 39, 2-3 (May 2000), 103–134.

[23] Bo Pang and Lillian Lee. 2008. Opinion Mining and Sentiment Analysis. *Found. Trends Inf. Retr.* 2, 1-2 (Jan. 2008), 1–135.

[24] Matthew Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep Contextualized Word Representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*. Association for Computational Linguistics, 2227–2237.

[25] M. Schuster and K.K. Paliwal. 1997. Bidirectional Recurrent Neural Networks. *Trans. Sig. Proc.* 45, 11 (Nov. 1997), 2673–2681.

[26] Richard Socher, Brody Huval, Christopher D. Manning, and Andrew Y. Ng. 2012. Semantic Compositionality through Recursive Matrix-Vector Spaces. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*. Association for Computational Linguistics, Jeju Island, Korea, 1201–1211.

[27] Xiang Zhang, Junbo Zhao, and Yann LeCun. 2015. Character-level Convolutional Networks for Text Classification. In *Proceedings of the 28th International Conference on Neural Information Processing Systems - Volume 1 (NIPS'15)*. MIT Press, Cambridge, MA, USA, 649–657.