Adversarial Training for Relation Extraction

Yi Wu
Computer Science Division
UC Berkeley
jxwuyi@gmail.com

David Bamman
School of Information
UC Berkeley
dbamman@berkeley.edu

Stuart Russell
Computer Science Division
UC Berkeley
russell@berkeley.edu

Abstract

Adversarial training is a mean of regularizing classification algorithms by generating adversarial noise to the training data. We apply adversarial training in relation extraction within the multi-instance multi-label learning framework. We evaluate various neural network architectures on two different datasets. Experimental results demonstrate that adversarial training is generally effective for both CNN and RNN models and significantly improves the precision of predicted relations.

1 Introduction

Despite the recent successes of deep neural networks on various applications, neural network models tend to be overconfident about the noise in input signals. *Adversarial examples* (Szegedy et al., 2013) are examples generated by adding noise in the form of small perturbations to the original data, which are often indistinguishable for humans but drastically increase the loss incurred in a deep model. *Adversarial training* (Goodfellow et al., 2014) is a technique for regularizing deep models by encouraging the neural network to correctly classify both unmodified examples and perturbed ones, which in practice not only enhances the robustness of the neural network but also improves its generalizability. Previous work has largely applied adversarial training on straightforward classification tasks, including image classification (Goodfellow et al., 2014) and text classification (Miyato et al., 2016), where the goal is simply predicting a single label for every example and the training examples are able to provide strong supervision. It remains unclear whether adversarial training could be still effective for tasks with much weaker supervision, e.g., distant supervision (Mintz et al., 2009), or a different evaluation metric other than prediction accuracy (e.g., F1 score).

This paper focuses on the task of *relation extraction*, where the goal is to predict the relation that exists between a particular entity pair given several text mentions. One popular way to handle this problem is the multi-instance multi-label learning framework (MIML) (Hoffmann et al., 2011; Surdeanu et al., 2012) with distant supervision (Mintz et al., 2009), where the mentions for an entity pair are aligned with the relations in Freebase (Bollacker et al., 2008). In this setting, relation extraction is much harder than the canonical classification problem in two respects: (1) although distant supervision can provide a large amount of data, the training labels are very noisy, and due to the multi-instance framework, the supervision is much weaker; (2) the evaluation metric of relation extraction is often the precision-recall curve or F1 score, which cannot be represented (and thereby optimized) directly in the loss function.

In order to evaluate the effectiveness of adversarial training for relation extraction, we apply it to two different architectures (a convoluational neural network and a recurrent neural network) on two different datasets. Experimental results show that even on this harder task with much weaker supervision, adversarial training can still improve the performance on all of the cases we studied.

2 Related Work

Neural Relation Extraction: In recent years, neural network models have shown superior performance over approaches using hand-crafted features in various tasks. Convolutional neural networks (CNN) are among the first deep models that have been applied to relation extrac-

Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing, pages 1778–1783
Copenhagen, Denmark, September 7–11, 2017.

tion (Santos et al., 2015; Nguyen and Grishman, 2015). Variants of convolutional networks include piecewise-CNN (PCNN) (Zeng et al., 2014), split CNN (Adel et al., 2016), CNN with sentence-wise pooling (Jiang et al., 2016) and attention CNN (Wang et al., 2016). Recurrent neural networks (RNN) are another popular choice, and have been used in recent work in the form of recurrent CNNs (Cai et al., 2016) and attention RNNs (Zhou et al., 2016). An instance-level selective attention mechanism was introduced for MIML by Lin et al. (2016), and has significantly improved the prediction accuracy for several of these base deep models.

Adversarial Training: Adversarial training (AT) (Goodfellow et al., 2014) was originally introduced in the context of image classification tasks where the input data is continuous. Miyato et al. (2015, 2016) adapts AT to text classification by adding perturbations on word embeddings and also extends AT to a semi-supervised setting by minimizing the entropy of the predicted label distributions on unlabeled data.

AT introduces an end-to-end and deterministic way of data perturbation by utilizing the gradient information. There are also other works for regularizing classifiers by adding random noise to the data, such as dropout (Srivastava et al., 2014) and its variant for NLP tasks, word dropout (Iyyer et al., 2015). Xie et al. (2017) discusses various data noising techniques for language models. Søgaard (2013) and Li et al. (2017) focus on linguistic adversaries.

3 Methodology

We first introduce MIML and then describe the base neural network models we consider:[1] piecewise CNN (Zeng et al., 2015) (PCNN) and bidirectional GRU (Cho et al., 2014) (RNN). We also utilize the *selective attention* mechanism in Lin et al. (2016) for both PCNN and RNN models. Adversarial training is presented at the end of this section.

3.1 Preliminaries

In MIML, we consider the set of text sentences $X = \{x_1, x_2, \ldots, x_n\}$ for each entity pair. Supposing we have R predefined relations (including *NA*) to extract, we want to predict the probability of each of the R relations given the mentions. Formally, for each relation r, we want to predict $P(r \mid x_1, \ldots, x_n)$.

Note that since an entity pair may have no relations, we introduce a special relation *NA* to the label set. Hence, we simply assume there will be *at least* one relation existing for every entity pair. During evaluation, we ignore the probability predicted for the *NA* relation.

[1] We primarily focus on effectiveness of AT. Other techniques in Sec. 2 are complementary to our focus.

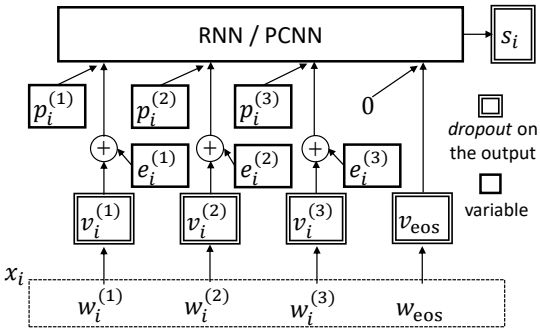

Figure 1: The computation graph of encoding a sentence x_i with adversarial training. e_i denotes the adversarial perturbation w.r.t. x_i. Dropout is placed on the output of the variables in the double-lined rectangles.

3.2 Neural Architectures

Input Representation: For each sentence x_i, we use pretrained word embeddings to project each word token into d_w-dimensional space. Note that we also need to include the entity position information in x_i. Here we introduce an extra feature vector $p_i^{(w)}$ for each word w to encode the entities' positions. One choice is the *position embedding* (Zeng et al., 2014): for each word w, we compute the relative distances to the two entities and embed the distances in two d_p-dimensional vectors, which are then concatenated as $p_i^{(w)}$. Position embedding introduces extra variables in the model and slows down the training time. We also investigate a simpler choice, *indicator encoding*: when a word w is exactly an entity, we generate a d_p-dimensional $\vec{1}$ vector and a $\vec{0}$ vector otherwise. In our experiments, position embedding is crucial for PCNN due to the spatial invariance of CNN. For RNN, position embedding helps little (likely because an RNN has the capacity of exploiting temporal dependencies) so we adopt indicator encoding instead.

Sentence Encoder: For a sentence x_i, we want to apply a non-linear transformation to the vector

representation of x_i to derive a feature vector $s_i = f(x_i;\theta)$ given a set of parameters θ. We consider both PCNN and RNN as $f(x_i;\theta)$.

For PCNN, inheriting the settings from (Zeng et al., 2014), we adopt a convolution kernel with window size 3 and d_s output channels and then apply piecewise pooling and ReLU (Nair and Hinton, 2010) as an activation function to eventually obtain a $3 \cdot d_s$-dimensional feature vector s_i.

For RNN, we adopt bidirectional GRU with d_s hidden units and concatenate the hidden states of the last timesteps from both the forward and the backward RNN as a $2 \cdot d_s$-dimensional feature vector s_i.

Selective Attention: Following Lin et al. (2016), for each relation r, we aim to softly select an attended sentence s_r by taking a weighted average of $s_1, s_2, \ldots, s_n$, namely $s_r = \sum_i \alpha_i^r s_i$. Here α^r denotes the attention weights w.r.t. relation r. For computing the weights, we define a *query vector* q_r for each relation r and compute $\alpha^r = \text{softmax}(u^r)$ where $u_i^r = \tanh(s_i)^\top q_r$. The query vector q_r can be considered as the embedding vector for the relation r, which is jointly learned with other model parameters.

Loss Function: For an entity pair, we compute the probability of relation r by $P(r \mid X;\theta) = \text{softmax}(As_r + b)$, where A is the projection matrix and b is the bias. For the multi-label setting, suppose K relations $r_1, \ldots, r_K$ exist for X. Simply taking the summation over the log probabilities of all those labels yields the final loss function

$$L(X;\theta) = -\sum_{i=1}^{K} \log P(r_i \mid X;\theta). \quad (1)$$

Dropout: For regularizing the parameters, we apply dropout (Srivastava et al., 2014) to both the word embedding and the sentence feature vector s_i. Note that we do not perform dropout on the position embedding p_i.

3.3 Adversarial Training

Adversarial training (AT) is a way of regularizing the classifier to improve robustness to small *worst-case* perturbations by computing the gradient direction of a loss function w.r.t. the data. AT generates continuous perturbations, so we add the adversarial noise at the level of the word embeddings, similar to Miyato et al. (2016). Formally, consider the input data X and suppose the word embedding of all the words in X is V. AT adds a small adversarial perturbation e_{adv} to V and optimizes the following objective instead of Eq.(1).

$$L_{\text{adv}}(X;\theta) = L(X + e_{\text{adv}};\theta), \text{ where} \quad (2)$$

$$e_{\text{adv}} = \arg\max_{\|e\| \le \epsilon} L(X + e;\hat{\theta}) \quad (3)$$

Here $\hat{\theta}$ denotes a fixed copy of the current value of θ. Since Eq.(3) is computationally intractable for neural nets, Goodfellow et al. (2014) proposes to approximate Eq.(3) by linearizing $L(X;\hat{\theta})$ near X:

$$e_{\text{adv}} = \epsilon g / \|g\|, \text{ where } g = \nabla_V L(X;\hat{\theta}). \quad (4)$$

Here V denotes the word embedding of *all* the words in X. Accordingly, in Eq. 4, $\|g\|$ denotes the norm of gradients over *all* the words from *all* the sentences in X. In addition, we do not perturb the feature vector p for entity positions. A visualization of the process is demonstrated in Fig. 1.

Dataset	#Rel	#Ent-Pair	#Mention	Sent-Len
NYT-Train	58	290429	577434	145
UW-Train	5	132419	546731	120

Table 1: Dataset statistics (#Rel includes *NA*).

4 Experiments

To measure the effectiveness of adversarial training on relation extraction, we evaluate both the CNN (PCNN) and RNN (bi-GRU) models on two different datasets, the NYT dataset (NYT) developed by Riedel et al. (2010) and the UW dataset (UW) by Liu et al. (2016). All code is implemented in Tensorflow (Abadi et al., 2016) and available at `https://github.com/jxwuyi/AtNRE`. We adopt Adam optimizer (Kingma and Ba, 2014) with learning rate 0.001, batch size 50 and dropout rate 0.5. For adversarial training, the only parameter is ϵ. In each of the following experiments, we fixed all the hyper-parameters of the base model, performed a binary search solely on ϵ and showed the most effective value of ϵ.

4.1 Datasets

The statistics of the two datasets are summarized in Table 1. We exclude sentences longer than *Sent-Len* during training and randomly split data for entity pairs with more than 500 mentions. Note that the number of target relations in these two datasets are significantly different, which helps

Recall	0.1	0.2	0.3	0.4	AUC
PCNN	0.667	0.572	0.476	0.392	0.329
PCNN-Adv	0.717	0.589	0.511	0.407	0.356
RNN	0.668	0.586	0.524	0.442	0.351
RNN-Adv	**0.728**	**0.646**	**0.553**	**0.481**	**0.382**

Table 2: Precisions of various models for different recalls on the NYT dataset, with best values in bold.

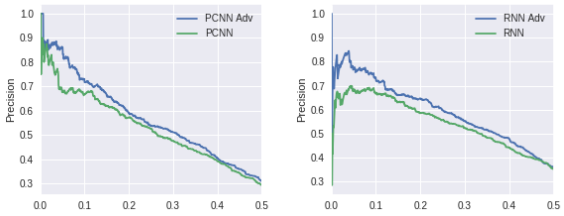

Figure 2: PR curves for PCNN (left) and RNN (right) on the NYT dataset with (blue) and without (green) adversarial training.

demonstrate the applicability of adversarial training on various evaluation settings.

Since the test set of the UW dataset only contains 200 sentences, we adopt a subset of the test set from the NYT dataset: all the entity pairs with the corresponding 4 relations in UW and another 1500 randomly selected *NA* pairs.

4.2 Practical Performances

The NYT dataset:

We utilize the word embeddings released by Lin et al. (2016), which has $d_w = 50$ dimensions. For model parameters, we set $d_e = 5$ (dimension of the entity position feature vector) and $d_s = 230$ (dimension of sentence feature vector) for PCNN and $d_e = 3$ and $d_s = 150$ for RNN. For adversarial training, we choose $\epsilon = 0.01$ for PCNN and $\epsilon = 0.02$ for RNN. We empirically observed that when adding dropout to the word embeddings, PCNN performs significantly worse. Hence we only apply dropout to s_i for PCNN. However, even with a dropout rate of 0.5, RNN still performs well. We conjecture that it is due to PCNN being more sensitive to input signals and the dimensionality of the word embedding ($d_w = 50$) being very small.

The precision-recall curves for different models on the test set are shown in Fig. 2. Since the precision drops significantly with large recalls on the NYT dataset, we emphasize a part of the curve with recall number smaller than 0.5 in the figure. Adversarial training significantly improves the precision for both PCNN and RNN models. We also show the precision numbers for some particular recalls as well as the AUC (for the whole PR curve) in Table 2, where RNN generally leads to better precision.

Recall	0.1	0.2	0.3	0.4	AUC
PCNN	0.765	0.717	0.713	0.677	0.576
PCNN-Adv	0.844	0.750	0.738	0.707	0.619
RNN	0.823	0.822	0.791	0.752	0.631
RNN-Adv	**0.929**	**0.878**	**0.850**	**0.779**	**0.671**

Table 3: Precisions of various models for different recalls on the UW dataset, with best values in bold.

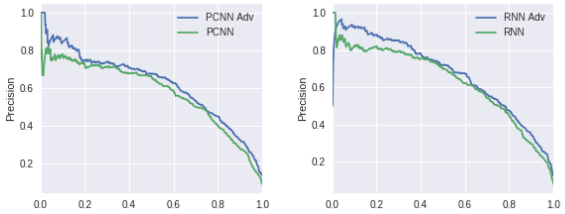

Figure 3: PR curves for PCNN (left) and RNN (right) on the UW dataset with (blue) and without (green) adversarial training.

The UW dataset:

We train a word embedding of $d_w = 200$ dimensions using Glove (Pennington et al., 2014) on the New York Times Corpus in this experiment. For model parameters, we set the entity feature dimension $d_e = 5$ and sentence feature dimension $d_s = 250$ for PCNN and $d_e = 3$ and $d_s = 200$ for RNN. For adversarial training, we choose $\epsilon = 0.05$ for PCNN and $\epsilon = 0.5$ for RNN. Since here word embedding dimension d_w is larger than that used for the NYT dataset, which implies that we now have word embeddings with larger norms, accordingly the optimal value of ϵ increases. The precision-recall curves on the test data are shown in Fig. 3, where adversarial training again significantly improves the precision for both models. The precision numbers for some particular recall values as well as the AUC numbers are demonstrated in Table 3. Similarly RNN yields superior performances on the UW dataset.

4.3 Discussion

CNN vs RNN: In the experiments, RNN generally produces more precise predictions than CNN due to its rich model capacity and also has high

robustness to input embeddings. The CNN, in contrast, has far fewer parameters which leads to much faster training and testing, which suggests a practical trade-off.

Notably, although the improvement under AUC by adversarial training are roughly the same for both RNN and CNN, the optimal ϵ value for RNN is always much larger than CNN. This implies that empirically RNN is more robust under adversarial attacks than CNN, which also helps RNN maintain higher precision as recall increases.

Choice of ϵ: When $\epsilon = 0$, the AT loss (Eq.(2)) degenerates to the original loss (Eq.(1)); when ϵ becomes too large, the noise can change the semantics of a sentence[2] and make the model extremely hard to correctly classify the adversarial examples.

Notably, the optimal value of ϵ is much smaller than the norm of the word embedding, which implies adversarial training works most effectively when only producing tiny perturbations on word features while keeping the semantics of sentences unchanged[3].

Connection to other approaches: Li et al. (2017); Xie et al. (2017) proposes linguistic adversaries techniques to enhance the robustness of the model by randomly changing the word tokens in a sentence. This explicitly modifies the semantics of a sentence. By contrast, adversarial training focuses on smaller and continuous perturbations in the embedding space while preserving the semantics of sentences. Hence, adversarial training is complementary to linguistic adversaries.

Acknowledgments

We would like to thank our anonymous reviewers for valuable discussions. We also appreciate the generous help from Angli Liu and Yankai Lin. This work is partially supported by the DARPA PPAML program, contract FA8750-14-C-0011.

References

Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, et al. 2016. Tensorflow: Large-scale machine learning on heterogeneous distributed systems. *arXiv preprint arXiv:1603.04467*.

Heike Adel, Benjamin Roth, and Hinrich Schütze. 2016. Comparing convolutional neural networks to traditional models for slot filling. In *NAACL HLT 2016, The 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, San Diego, California, USA, June 12 - June 17, 2016*.

Kurt Bollacker, Colin Evans, Praveen Paritosh, Tim Sturge, and Jamie Taylor. 2008. Freebase: a collaboratively created graph database for structuring human knowledge. In *Proceedings of the 2008 ACM SIGMOD international conference on Management of data*, pages 1247–1250. ACM.

Rui Cai, Xiaodong Zhang, and Houfeng Wang. 2016. Bidirectional recurrent convolutional neural network for relation classification. *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics. Association for Computational Linguistics*.

Kyunghyun Cho, Bart Van Merriënboer, Dzmitry Bahdanau, and Yoshua Bengio. 2014. On the properties of neural machine translation: Encoder-decoder approaches. *arXiv preprint arXiv:1409.1259*.

Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. 2014. Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572*.

Raphael Hoffmann, Congle Zhang, Xiao Ling, Luke Zettlemoyer, and Daniel S Weld. 2011. Knowledge-based weak supervision for information extraction of overlapping relations. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies-Volume 1*, pages 541–550. Association for Computational Linguistics.

Mohit Iyyer, Varun Manjunatha, Jordan L Boyd-Graber, and Hal Daumé III. 2015. Deep unordered composition rivals syntactic methods for text classification. In *Proceedings of ACL*.

Xiaotian Jiang, Quan Wang, Peng Li, and Bin Wang. 2016. Relation extraction with multi-instance multi-label convolutional neural networks. *Proceedings of COLING*.

Diederik Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Yitong Li, Trevor Cohn, and Timothy Baldwin. 2017. Robust training under linguistic adversity. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics*.

[2]When ϵ is large enough, e.g., comparable to the norm of input embeddings, and if we add the perturbation to word w and consider its nearest unperturbed word embedding in the embedding space, the nearest word will be different from the original word w. This implicitly changes the content of a sentence.

[3]The nearest neighbor of word w remains unchanged after w being perturbed with the optimal ϵ.

Yankai Lin, Shiqi Shen, Zhiyuan Liu, Huanbo Luan, and Maosong Sun. 2016. Neural relation extraction with selective attention over instances. In *Proceedings of ACL*, volume 1, pages 2124–2133.

Angli Liu, Stephen Soderland, Jonathan Bragg, Christopher H Lin, Xiao Ling, and Daniel S Weld. 2016. Effective crowd annotation for relation extraction. In *Proceedings of the NAACL-HLT*.

Mike Mintz, Steven Bills, Rion Snow, and Dan Jurafsky. 2009. Distant supervision for relation extraction without labeled data. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP: Volume 2-Volume 2*, pages 1003–1011. Association for Computational Linguistics.

Takeru Miyato, Andrew M Dai, and Ian Goodfellow. 2016. Adversarial training methods for semi-supervised text classification. *arXiv preprint arXiv:1605.07725*.

Takeru Miyato, Shin-ichi Maeda, Masanori Koyama, Ken Nakae, and Shin Ishii. 2015. Distributional smoothing with virtual adversarial training. *arXiv preprint arXiv:1507.00677*.

Vinod Nair and Geoffrey E Hinton. 2010. Rectified linear units improve restricted boltzmann machines. In *Proceedings of the 27th international conference on machine learning (ICML-10)*, pages 807–814.

Thien Huu Nguyen and Ralph Grishman. 2015. Relation extraction: Perspective from convolutional neural networks. In *Proceedings of NAACL-HLT*, pages 39–48.

Jeffrey Pennington, Richard Socher, and Christopher D Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of EMNLP*, volume 14, pages 1532–1543.

Sebastian Riedel, Limin Yao, and Andrew McCallum. 2010. Modeling relations and their mentions without labeled text. *Machine learning and knowledge discovery in databases*, pages 148–163.

Cicero Nogueira dos Santos, Bing Xiang, and Bowen Zhou. 2015. Classifying relations by ranking with convolutional neural networks.

Anders Søgaard. 2013. Part-of-speech tagging with antagonistic adversaries. In *Proceedings of ACL*, pages 640–644.

Nitish Srivastava, Geoffrey E Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: a simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*, 15(1):1929–1958.

Mihai Surdeanu, Julie Tibshirani, Ramesh Nallapati, and Christopher D Manning. 2012. Multi-instance multi-label learning for relation extraction. In *Proceedings of the 2012 joint conference on empirical methods in natural language processing and computational natural language learning*, pages 455–465. Association for Computational Linguistics.

Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. 2013. Intriguing properties of neural networks. *arXiv preprint arXiv:1312.6199*.

Linlin Wang, Zhu Cao, Gerard de Melo, and Zhiyuan Liu. 2016. Relation classification via multi-level attention cnns. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics. Association for Computational Linguistics*.

Ziang Xie, Sida I Wang, Jiwei Li, Daniel Lévy, Aiming Nie, Dan Jurafsky, and Andrew Y Ng. 2017. Data noising as smoothing in neural network language models. *arXiv preprint arXiv:1703.02573*.

Daojian Zeng, Kang Liu, Yubo Chen, and Jun Zhao. 2015. Distant supervision for relation extraction via piecewise convolutional neural networks. In *Proceedings of EMNLP*, pages 1753–1762.

Daojian Zeng, Kang Liu, Siwei Lai, Guangyou Zhou, Jun Zhao, et al. 2014. Relation classification via convolutional deep neural network. In *Proceedings of COLING*, pages 2335–2344.

Peng Zhou, Wei Shi, Jun Tian, Zhenyu Qi, Bingchen Li, Hongwei Hao, and Bo Xu. 2016. Attention-based bidirectional long short-term memory networks for relation classification. In *Proceedings of The 54th Annual Meeting of the Association for Computational Linguistics*, page 207.