RDSGAN: Rank-based generative adversarial training for distant supervision relation extraction

Anonymous COLING submission

Abstract

Distant supervised relation extraction has been widely used for constructing large scale knowledge bases, but suffers from the wrong label problem heavily. In this paper, we propose a generative neural model called RDSGAN (Rank-based DSGAN), which can generate valid instances for distant supervised relation extraction. To address the false positive problem, our method learns the distribution of true positive instances and is able to generate massive valid instances according to the given triplet. Attention Mechanism Experimental results on the New York Times dataset show that, our model promotes distant supervised relation extraction and achieves state-of-art results compared with baselines.

1 Introduction

Despite the recent successes of deep neural networks and multi-instance learning (Lin et al., 2016; Du et al., 2018) on denoising operation with distant supervision (Mintz et al., 2009), neural network models tend to be overconfident about the noise in input signals. Various attention mechanisms (Han et al., 2018; Gao et al., 2019) have been explored for calculating attention weights. However, soft attention mechanism usually assign non-zero weights to noisy sentences, which do not completely eliminate noisy data. (Qin et al., 2018b) and (Qin et al., 2018a) then remove false positive instances from positive set with hard decision and use reinforcement learning. However, the dataset may lose useful information contained in removed instances and does not take into account a large number of false negative instances. Based on this consideration, learning the distribution of valid instances[1] in the dataset and then automatically generating training instances can be seen as a better way to solve the wrong label problem.

In this paper, we propose a generative neural model for distant supervised relation extraction to solve the false positive problem, which learn the distribution of valid instances based on adversarial learning (Goodfellow et al., 2014) and then generate training samples according to the given triplet. As illustrated in Figure 1, given a triplet *(head, relation, tail)*, the goal of our model is to generate a vector sequence representing an instance, which contains less noise. We firstly close the generator and pretrain the other modules in order to obtain basic ability of relation classification. Then, we utilize adversarial learning to train our model. In first step, we fix the discriminator and only train the generator. The decoder-based generator generates an instance and incorporates it into a bag. Then the ranking module ranks the mixed instances based on attention mechanism and produces the bag representation for relation classification. The target of the generator is to generate instances which obey data distribution. In second step, we fix the generator and use the generated instances to train the relation classifier. After repeating the above two steps several epoches, the generator can finally learn to generate valid instances.

The major contributions in this study are two-fold:

- To address the false positive problem, our method learns the distribution of true positive instances and then generate valid instances instead of directly using soft attention, which can generate massive valid instances according to the given triplet.

[1] Valid instances include true positive and true negative instances

- Experimental results on the widely used New York Times (NYT) dataset show that our model, generating massive valid instances according to a given triplet, achieves state-of-art performance in relation extraction compared with baselines.

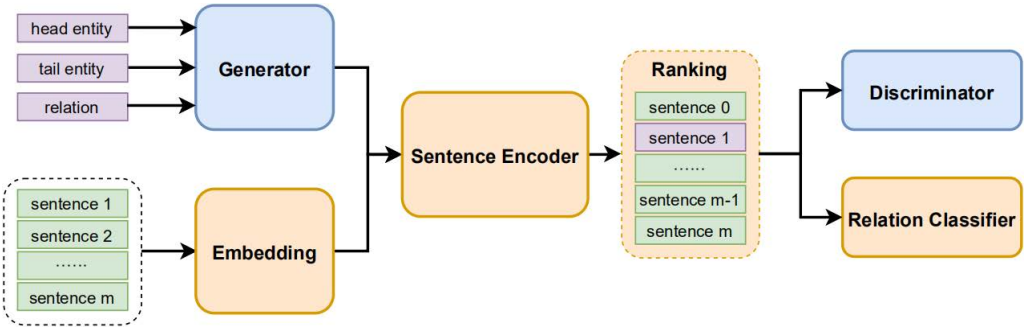

Figure 1: The architecture of our model RDSGAN. The left part is the generator consist of a sentence encoder and a deconvolutional network, which merge the features of the sentences in a bag and the feature of the generated instance. The right part is the discriminator consist of a ranking module and feed-forward networks, which firstly rank the mixed feature vectors and then perform relation classification task and discrimination task.

2 Related Work

Generative Adversarial Training: In recent years, neural network models have shown superior performance over approaches using hand-crafted features in various tasks. Generative Adversarial Networks (GANs) (Goodfellow et al., 2014) and their variants such as Conditional GAN (Mirza and Osindero, 2014), InfoGAN (Chen et al., 2016) and ACGAN (Odena et al., 2017) appear as very prospective techniques for data generation, especially in the Computer Vision domain. Text generation has not achieved equal success due to its discrete property in words. However, recent studies have proposed several approaches to deal with this problem. A reinforcement learning (RL) algorithm is used to upgrade the generator with reward signals instead of back-propagation in (Yu et al., 2017)

Attention Mechanism: Attention mechanism is generally considered as an effective way to reduce noise in distant supervised relation extraction. (Lin et al., 2016) build sentence-level attention over multiple instances in order to dynamically reduce the weights of those noisy instances. (Zhang et al., 2018) explore the attention-based capsule networks in a multi-instance multi-label learning framework for relation extraction. (Han et al., 2018) proposes a hierarchical attention schema which provides coarse-to-fine granularity to better identify valid instances.

Although methods concerning attention mechanism have made great progress in distant supervised neural relation extraction, they mainly focus on building soft attention on multiple instances, which cannot completely eliminate the noise. In addition, roughly redistributing the dataset may lose useful information contained in removed instances.

3 Methodology

In this section, we introduce our generative neural model which addresses the false positive problem via learning the distribution of valid instances, and then generates instances according to the given triplet. Next, we will first discuss the common relation classifier, followed by the part of generator and discriminator, then introduce the loss functions of our model, and finally discuss the details of the training strategy.

3.1 Common Relation Classifier

As is illustrated in Figure 1, the common relation classifier part consists of embedding layer, sentence encoder and ranking module as well as the relation classifier. The embedding layer maps each sentence

into a vector sequence, and the sentence encoder is used to extract feature from sentence instances, and the ranking module is utilized to rank instances in a bag according to the noise they contain. Finally, a representation of the bag is fed into the relation classifier for relation classification.

3.1.1 Embedding Layer

The embedding layer transforms a sentence into a vector sequence in which each word is represented as a dense vector. Following (Zeng et al., 2015), we also use position embeddings to emphasize the import of position information so the output of a word contains both word embeddings and position embeddings.

Suppose the dimension of word embeddings is d_w, the dimension of each position embeddings is d_p, hence the dimension of word w in sentence s is $d = d_w + 2d_p$. Given a sentence s consists of L words $s = \{w_1, w_2, \cdots, w_L\}$, we denote the vector sequence of it as $\mathbf{s} = \{\mathbf{w}_1, \mathbf{w}_2, \cdots, \mathbf{w}_L\}$, where L is the length of each sentence, d_s is the dimension of the encoded sentence.

3.1.2 Sentence Encoder

Neural network such as convolutional neural networks (CNNs) and recurrent neural networks (RNNs) are utilized to overcome error propagation caused by traditional feature extracting tools. Following (Zeng et al., 2014), we utilize CNNs to extract features from sentence instances in experiments. Hence, the convolution operation of the convolutinal layer is denoted as:

$$\mathbf{p} = \mathbf{W}_e\mathbf{q} + \mathbf{b}_1 \tag{1}$$

where $\mathbf{b}_1$ is the bias vector. And then the final representation $\mathbf{x} \in R^{d_s}$ of a sentence is calculated as:

$$\mathbf{x} = \text{RELU}(\max(\mathbf{p})) \tag{2}$$

where max is the max-pooling operation, RELU is rectified linear unit used for activation function.

It should be noted that we concentrate on generating valid instances for relation extraction which are independent of the specific encoder. The generated instances can also promote other encoders.

3.1.3 Ranking Module

We build the ranking module based on attention mechanism, which ranks the instances according to their attention weights. The attention weight for sentence $\mathbf{x}_i$ is defined as:

$$\alpha_i = \frac{\exp(e_i)}{\sum_k \exp(e_k)} \tag{3}$$

where e_i is refered as a query-based function which scores how well the input sentence $\mathbf{x}_i$ and the predicting relation $\mathbf{r}$ matches. Following lin2016neural, we also select the bilinear form in different alternatives:

$$e_i = \mathbf{x}_i\mathbf{A}\mathbf{r} \tag{4}$$

where $\mathbf{A}$ is a weighted diagonal matrix which has been used in the deconvolutional network, and $\mathbf{r}$ is the query vector associated with relation r which indicates the representation of relation r.

3.1.4 Relation Classifier

Given a bag of M containing m sentences related to entity pair (h, t), the representation of M can be calculated as:

$$\mathbf{q} = \sum_{i=1}^{m} \alpha_i \mathbf{x}_i \tag{5}$$

Then the conditional probability of (h, t) expressing relation r is defined as:

$$p(r|M; \Theta) = \frac{\exp(o_r)}{\sum_{i=1}^{N_r} \exp(o_i)} \tag{6}$$

where Θ represents all parameters. N_r is the total number of relation classes, and o_r is the scores for relation r:

$$\mathbf{o} = \mathbf{W}_r\mathbf{q} + \mathbf{b}_2 \tag{7}$$

where $\mathbf{W}_r$ is the weight matrix and $\mathbf{b}_2$ is a bias vector.

3.2 Generative Adversarial Training

As is shown in Figure 1, the generator and discriminator part consists of a generator and a discriminator. The encoder-based generator inputs a triplet and output a vector sequence representing a valid sentence instance. The discriminator attempts to identify whether an instance comes from the real data or not. We argue that the generator can produce valid instances for relation extraction based on the adversarial learning method.

3.2.1 Generator

The goal of the generator is to generate a vector sequence representing a valid instance of the given triplet. Hence, given the triplet of (h, r, t), we first map h and t into vectors via their word embeddings and map r via a relation matrix $\mathbf{A} \in R^{N_r \times d_s}$, i.e. $\mathbf{e}_r = \mathbf{A}\mathbf{r}$, where N_r is the number of all relation classes, and d_s is the dimension of sentence embedding, $\mathbf{r}$ is the query vector associated with relation r. The input of the generator is the sum of the three vectors:

$$\mathbf{z} = \mathbf{e}_h + \mathbf{W}_g\mathbf{e}_r + \mathbf{e}_t \tag{8}$$

We propose a decoder-based generator to generate vector sequence representing an instance. In detail, we utilize LSTM for the decoder and the generating process can be formulated as:

$$\mathbf{h}_{i+1} = LSTM(\mathbf{h}_i) \tag{9}$$

where $\mathbf{h}_i \in R^d$ is the hidden vector of the LSTM and $\mathbf{h}_0 = \mathbf{z}$. The generating process goes on until it reaches the aligned length L.

3.2.2 Discriminator

The goal of the discriminator is to identify whether an instance comes from the real data or the generator. The generator attempts to generate instances which deceive the discriminator, and the discriminator tries its best to identify the generated instances. For each instance in a bag, the discriminator calculates its probability of coming from the real data as follows:

$$D(\mathbf{x}_i) = p(\mathbf{x}_i \sim D_R) \tag{10}$$

where D_R is the distribution of the real data.

After the generating process, we obtain a sentence bag $\mathbf{X} = \{\mathbf{x}_0, \mathbf{x}_1, \cdots, \mathbf{x}_m\}$ where $\mathbf{x}_0$ is the generated instances and the others are real data, then we feed the sentence bag into the discriminator, which should identify each sentence correctly. Furthermore, the generated instance must be ranked in the top-k in order to make the generator learn the distribution of true positive instances excluding that of false positive instances.

3.3 Loss Function

The objective function of our model consists of three part, the rank loss, the classification loss and the discrimination loss. When discussing, we assume that there are N_b bags in training data and the i-th bag contains m_i instances.

3.3.1 Relation Classification Loss

We define the loss of relation classification using cross-entropy as follows:

$$\mathcal{L}_1 = \sum_{i=1}^{N_b} = \log p(r_i|M_i; \Theta) \tag{11}$$

where M_i is the i-th bag in the training data.

3.3.2 Rank Loss

In order to let the generator learn the distribution of true positive instances, we attempt to make the generated instances rank in the top-k. Hence, the rank loss of the generated instance in a bag can be calculated as follows:

$$L_{rank}^{G} = \begin{cases} 0 & \text{rank}(\mathbf{x}_0) \leq k \\ \frac{\text{rank}(\mathbf{x}_0) - k}{m} & \text{rank}(\mathbf{x}_0) > k \end{cases} \tag{12}$$

where $\text{rank}(\mathbf{x}_i)$ is the ascending order of the attention weight of $\mathbf{x}_i$. k is a hyperparameter and m is the number of instances in the bag.

The total rank loss can be calculated as the sum of the rank loss of each bag:

$$\mathcal{L}_2 = \sum_{N_b} L_{rank}^{G} \tag{13}$$

3.3.3 Discrimination Loss

To learn the distribution of the real data, we design the discriminator to identify whether an instance comes from the real data or the generator. Hence, the objective of the discriminator can be formulated as the following cross-entropy loss function:

$$\mathcal{L}_3 = \sum_{N_b} \sum_{M_i} (\log D(\mathbf{x}_i) + \log(1 - D(\mathbf{x}_i)) \tag{14}$$

The final objective function is defined as the sum of the three loss function:

$$\mathcal{L} = \mathcal{L}_1 + \lambda_2 \mathcal{L}_2 + \lambda_3 \mathcal{L}_3 \tag{15}$$

where $\lambda_2 > 0, \lambda_3 > 0$ and they are the weights of $\mathcal{L}_2$ and $\mathcal{L}_3$.

3.4 Training Strategy

The entire training process consists of pre-training and alternating training of the generator and discriminator. In order to make the generator learn the distribution of the real data, we first pretrain the common relation classifier part. Then, the remaining training process consists of alternating training of the generator and discriminator. In first step, we fix the discriminator and only train the generator. The decoder generate an instance and incorporate it into the bag for encoding. Then the ranking module rank the mixed instances based on attention mechanism and produce the final bag representation used for relation classification. The goal of the generator is to generate a vector sequence representing an instance which obey the data distribution and rank in the top-k. In second step, we fix the generator and use the generated instances to train the discriminator and other modules. By feeding generated instances rather than real data into the network, we believe that the ranking module and classifier can remember valid instances instead of invalid ones.

After repeating the above two steps several times, the generator can finally learn to generate valid instances.

4 Experiments

In this section, we firstly introduce the dataset and evaluation metrics we used, followed by the experimental settings and results. Finally, we present analysis of the results.

4.1 Dataset

We evaluate our model on NYT dataset developed by (Riedel et al., 2010), which aligns Freebase relations with the NYT corpus. The aligned sentences from years 2005-2006 of the NYT corpus are regarded as training data and those from year 2007 are for testing. Its entity mentions are found by Stanford named entity tagger and linked to Freebase. The dataset contains 53 relations including no relation "NA". There are 522,611 sentences linked to 281,270 entity pairs for training and 172,448 sentences linked to 96,678 entity pairs for testing.

4.2 Emperimental Settings

In experiments, we adopt stochastic gradient descent (SGD) for minimizing the objective function, and dropout strategy (Hinton et al., 2012) to prevent overfitting.In our experiments, we select the window size among $\{3, 4, 5\}$, the number of feature maps or filters among $\{128, 230, 256\}$, batch size among $\{32, 64, 160\}$, the learning rate among $\{0.1, 0.25, 0.5\}$.

4.3 Held-out Evaluation on NYT dataset

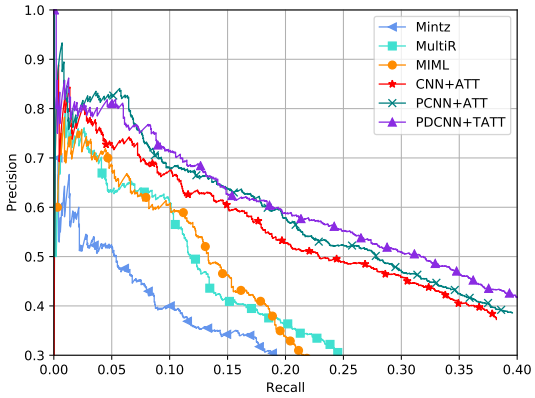

Figure 2: Comparison of aggregate precision/recall between our model and other models.

We compare our proposed RDSGAN framework with PCNN on NYT-Freebase dataset. Similar to previous works (Zeng et al., 2015; Lin et al., 2016), we evaluate the RDSGAN framework using the held-out evaluation. In 2, experiment results indicate that AC-GAN consistently outperforms PCNN. Our proposed data augmentation method improves the AUC of PR curve by 7.66% compared to PCNN. This is mainly due to the fact that using RDSGAN, the generator can consistently provide high-quality relational sentences.

5 Conclusion

In this paper, we propose an generative neural model for distant supervised relation extraction. RDSGAN learns the distribution of true positive instances and is able to generate massive valid instances according to the given triplet, which addresses the false positive problem. Experimental results on the New York Times dataset shows that, RDSGAN framework improves distant supervised relation extraction and achieves state-of-art performance.

Acknowledgements

References

Xi Chen, Yan Duan, Rein Houthooft, John Schulman, Ilya Sutskever, and Pieter Abbeel. 2016. Infogan: Interpretable representation learning by information maximizing generative adversarial nets. In *Advances in neural information processing systems*, pages 2172–2180.

Jinhua Du, Jingguang Han, Andy Way, and Dadong Wan. 2018. Multi-level structured self-attentions for distantly supervised relation extraction. *arXiv preprint arXiv:1809.00699*.

Tianyu Gao, Xu Han, Zhiyuan Liu, and Maosong Sun. 2019. Hybrid attention-based prototypical networks for noisy few-shot relation classification.

Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. 2014. Generative adversarial nets. In *Advances in neural information processing systems*, pages 2672–2680.

Xu Han, Pengfei Yu, Zhiyuan Liu, Maosong Sun, and Peng Li. 2018. Hierarchical relation extraction with coarse-to-fine grained attention. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2236–2245.

Geoffrey E Hinton, Nitish Srivastava, Alex Krizhevsky, Ilya Sutskever, and Ruslan R Salakhutdinov. 2012. Improving neural networks by preventing co-adaptation of feature detectors. *arXiv preprint arXiv:1207.0580*.

Yankai Lin, Shiqi Shen, Zhiyuan Liu, Huanbo Luan, and Maosong Sun. 2016. Neural relation extraction with selective attention over instances. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1, pages 2124–2133.

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.

Mehdi Mirza and Simon Osindero. 2014. Conditional generative adversarial nets. *arXiv preprint arXiv:1411.1784*.

Augustus Odena, Christopher Olah, and Jonathon Shlens. 2017. Conditional image synthesis with auxiliary classifier gans. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 2642–2651. JMLR. org.

Pengda Qin, Weiran Xu, and William Yang Wang. 2018a. Dsgan: Generative adversarial training for distant supervision relation extraction.

Pengda Qin, Weiran Xu, and William Yang Wang. 2018b. Robust distant supervision relation extraction via deep reinforcement learning. *arXiv preprint arXiv:1805.09927*.

Sebastian Riedel, Limin Yao, and Andrew McCallum. 2010. Modeling relations and their mentions without labeled text. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 148–163. Springer.

Lantao Yu, Weinan Zhang, Jun Wang, and Yong Yu. 2017. Seqgan: Sequence generative adversarial nets with policy gradient. In *Thirty-First AAAI Conference on Artificial Intelligence*.

Daojian Zeng, Kang Liu, Siwei Lai, Guangyou Zhou, Jun Zhao, et al. 2014. Relation classification via convolutional deep neural network.

Daojian Zeng, Kang Liu, Yubo Chen, and Jun Zhao. 2015. Distant supervision for relation extraction via piecewise convolutional neural networks. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1753–1762.

Ningyu Zhang, Shumin Deng, Zhanlin Sun, Xi Chen, Wei Zhang, and Huajun Chen. 2018. Attention-based capsule networks with dynamic routing for relation extraction. *arXiv preprint arXiv:1812.11321*.