Investigation and Enhancement of Relation Extraction with Attention-based Neural Networks

Lizi Chen
Department of Computer Science
New York University
New York, NY 10012
lc3397@nyu.edu

Guangyu Zhang
Department of Computer Science
New York University
New York, NY 10012
guangyu.zhang@nyu.edu

Abstract

Natural language text is typically very unstructured, which makes many different machine learning applications difficult. A knowledge graph created from relation extraction techniques can be used to model the relationships of different entities in a body of text. These graphs create a lot of value for other natural language applications such as question answering and reading comprehension. One of the most representative approaches in relation extraction to use the supervised paradigm in neural networks, which uses selected sets of features in the form of vectors; for example, Part-of-Speech (POS) tagging feature, and/or syntactic parsing. To identify the relations between pairs of entities in a diverse text data, we combine fundamental features and sentence-level information. One can look up vector forms of all words in the sentence in the form of vectors, then find the target entities in the sentence by the given training entity pair, in addition to all the informations of the sentence learned by the neural network. These vectors are then work as a whole (*i.e.,* concatenation) feed to a softmax classifier to to predict the according marked relationship label.

1 Introduction

Traditionally, before the recent upsurge in Deep Learning, the state-of-art method for relation classification are fundamentally based on statistical machine learning. To confront such task directly from a statistical point of view results in propagation of errors from previously derived existing NLP systems. There have been publications that demonstrate the capability of these statistical methods; i.e., POS(Part-of-Speech) Tagging, named entity tags, to extract entities relations and reached an precision of 67.6% (Mintz et al., 2009). In recent years, Deep Learning has seen boom in popularity for NLP problems with its ability to model complicated features, resolve ambiguous text, as well as ensure multiple lexical and sentence level features extraction.

Based on revising recent papers; in this experiment, we create program which consumes data into knowledge graph with both available and handcraft datasets. We provide analytical insight into the model as well as tweak the parameters, analyze the existing models, and further improve on selective ones.

2 Related Works

2.1 Distant Supervision

Adapt Distance Supervision
(Mintz et al., 2009) has an assumption such that if two entities participate in a relation, any sentence that contain those two entities might express that relation, while in reality may result in non-trivial

noises. The proposed distant supervision method generates training data automatically via aligning documents with known knowledge base. Features that can be extracted successfully includes sequence of words, POS tags, syntactic (dependency parser) features and performed named entity tagging by Stanford four-class named entity tagger. This model is reported to be able to extract 10k instances of 102 relations at a precision of 67.6%. In addition, (Riedel et al., 2010) alleviates noises via deciding whether two entities are related and also mentioned in a given sentence, and then applying "constraint-driven semi-supervision" to train.

Sentence Level Features
Word embedding has put many features regarding statistical NLP into consideration such as Part-Of-Speech Tagging, word segmentation, and dependency parsing. Lexical and sentence level features; however, can be devised as a cue for making the relation recognition decision. Sentence level features are presented via a max-pooled neural network. Words in a sentence are represented as word features and position features.

i). *Word features* includes vector representation of the word and the vector representation of the word in the sentence. A sentence of length L can be represented as an array of vectors, $(x_s, x_1, x_2, ..., x_e)$. To enrich the feature of the word itself in a context, we can choose size $w = 3$, and add the following vector to the sentence level encoding.

$$\{[x_s, x_1, x_2], [x_1, x_2, x_3], [x_2, x_3, x_4], ... [x_{23}, x_{24}, x_e]\}$$

ii). w can only be a very limited amount for a richer word features because it includes all w words sequentially indexed in a sentence. The introduction of *position features* walks around the complicated structure in a sentence by focusing on just the entities. Each token in a sentence has two vector numbers corresponding to the entities. For example, the word {wisconsin} in the example has distance vectors d_1 and d_2 with respect to the relative distance of entity words $[aaron \quad hohlbein]_{e_1}$ and $[soccer \quad player]_{e_2}$, $[d_1, d_2] = [-8, 3]$

iii). *Lexical Features* in addition to the previous two, can provide more information about the entities in the whole sentence. In an sample training sentence, *" ###BEGIN### aaron hoblbein born ... currently without a club . ###END### ". Entity 1 and Entity 2* are the pair of input entities, both the ones on the left and right of them are also parsed. A list of hypernyms of nouns in the sentences from WordNet is also parsed for additional lexical features.

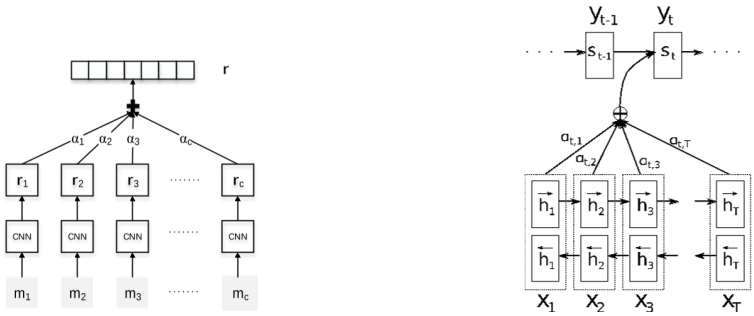
			H1			H2			
###BEGIN###			aaron	hohlbein	born	august			
Left to Entity_1			Entity_1			Right to Entity_1			
			H3		H4				
16	,	1986	in	middleton	,	wisconsin	is	an	
H5			H6						
american			soccer	player	who			is	
Left to Entity_2			Entity_2						
			H7						
currently	without	a	club	.	###END###				
			Right to Entity_2						

Figure 1: Lexical Feature Positions (in blue)

However, distant supervision has an assumption that if two entities appear in a sentence, then every sentence containing the same pair of entities ~~will~~ show the same relation. This results from the aligning source sentence to a knowledge base as supervision for extracting relation, and is a very strong relation. Thus, we will introduce deep learning methods.

3 Attention-based Neural Networks

Previous states of the art results rely on lexical features, while important information in a sentence can appear anywhere, and may be ignored by these manual feature selection models.



(a) The architecture of sentence-level attention-based CNN. Used in Yankai Lin et al 2016

(b) Classical attention mechanism used in Dzmitry arXiv:1409.0473

Figure 2: Comparison between two attentions.

3.1 Attention on the Entities

(Bahdanau et al., 2014) Breakthroughs neural machine translation model with attention mechanism, which evolves great success to more broader NLP tasks. At a high-level view in Figure 2a, the recurrent networks receives two fixed-size vectors as additional inputs at each output position t, which are (1) embedding of the previously generated output symbol e_{t-1}, and (2) encoding a "view" of the input matrix Blunsom (2017). To generate the fixed-size vectors at t, Bahdanau et al. proposed to compute a context vector c_i as a weighted sum of annotations h_i "containing information about the whole input sequence with a strong focus on the parts surrounding the i-th word of the input sequence", ***i.e.,***

$$c_i = \sum_{j=1}^{T_x} \alpha_{ij} h_j,$$

where

$$\alpha_{ij} = \frac{\exp(e_{ij})}{\sum_{k=1}^{T_x} \exp(e_{ij})},$$

is called *attention*; reflecting the importance of the annotation h_j with respect to the previous hidden state s_{i-1} in deciding the next state s_i, and generating y_i.

Also, $e_{ij} = a(s_{i-1}, h_j)$ is an alignment model scoring "how well the inputs around position j and the output at position i match" (Bahdanau et al., 2014).

3.2 Piecewise CNN

CNN models were introduced to relation extraction since 2014, e.g., the basic CNN model (Zeng et al., 2014), Piecewise CNN (Zeng et al., 2015), CNN with multiple kernels (Nguyen and Grishman, 2015), as well as the use of attention (Lin et al., 2016). Compare the tradeoffs between these model architectures and compare with non-deep learning baselines.

(Lin et al., 2016) improved the Distant supervised relation extraction method with additional attention mechanism over its previous work, as in 3. The improved version has CNN for semantic feature extraction and a sentence-level attention for the purpose of reducing wrong labeling weights.

Convolution
With all the features extraction in vector form as mentioned above, we need to utilize all these local features to predict a relation between a pair of entities globally. In neural networks, deep learning merges all the features as we expected. *i. Linear Transformation:*

$$Z_t = W_1 \times X$$

where $X \in \mathbb{R}^{(n_0 \times t)}$, is the concatenation of all the feature vectors. $n_0 = w \times n$, where n is a hyper-parameter corresponding to the dimension of feature vector. t is the number of nouns in a

sentence, including the entity pair. $W_1 \in R^{n_1 \times t}$ where n_1 refers to the number of input nodes of the first layer in the network.

$$Z_{max}{}^{i} = argmax(Z, i)$$

For i represents the $i-th$ row of the transformed matrix, We select the feature that has the greatest value in each row for the first layer input. This results in a uniformed length of vector, n_1

ii. Non-Linear Layer:

$$G = [ReLU, \tanh, Sigmoid]\,(W_2 \cdot m)$$

W_2 is just another linear transformation, $W_2 \in R^{n_2 \times n_1}$, where n_2 is the number of input nodes of another layer.
$[ReLU, Tanh, Sigmoid]$ We experiment with many non-linear functions and results in the fact that hyperbolic tanh works better as an activation function due to the fact that:

$$\frac{d}{dx}\tanh(x) = 1 - \tanh^2 x$$

(Lin et al., 2016) improved the Distant supervised relation extraction method with additional attention mechanism over its previous work, as in Figure 3. The improved version has CNN for semantic feature extraction and a sentence-level attention for the purpose of reducing wrong labeling weights.

Due to the fact that vector representation from previous step has all different size, max-pooling is required to obtain a fixed-sized vector for the input sentence. However, such single max pooling technique reduces the size of the hidden layers too rapidly and coarsely so that it does not fully retrieve the fine-grained details in relation extraction task. Also, the structural information of two target entities is not considered in a single max pooling. Due to the fact that entities appear in the context in different position; which in turn split the whole text into different structural regions, the introduction of piece-wise max pooling can take care of the previous concerns and missing. Instead of a single max value from the previous layer, piece-wise max pooling is applied to each segment of the text and returns the max for each part:

The input into a convolution layer is split by the two target entities into three parts:

$$\{s_{left}, s_{middle}, s_{right}\}$$

For the output of such layer, s_1 is fed into the piece-wise max pooling function as we have the final concatenated vector for the next non-linear function:

$$O_i = maxPool(s_{ileft}) + maxPool(s_{imiddle}) + maxPool(s_{iright})$$

3.3 Bidirectional Recurrent Networks

Since CNN is not suitable for persisting sequences information, and may thus lose long-term semantic features, (Zhang et al., 2015) proposed Bidirectional Long-Short-Term Memory networks for modeling relation classifications, with only embedding word vectors as input sequences had them achieved the-state-of-the-art result. They have also experimented with lexical features, dependency parsers as additional input vectors, and boosted the performance slightly. In contrast to these complex manual features, (Zhou et al., 2016) leverages four position indicators of two entities in a sentence, $\langle e1\rangle, \langle e1\rangle, \langle e2\rangle, \langle e2\rangle$ as single words, and also embedded as word vectors.

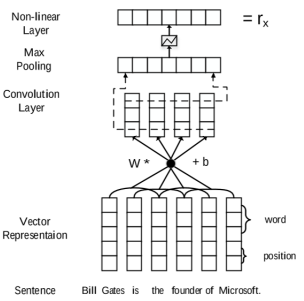


Figure 3: The architecture of CNN/PCNN used for sentence encoder

The model we experimented[1] has very similar structure in (Zhou et al., 2016) as Figure 4, other than adapting GRU cells instead of LSTM. The architecture of the model is:

1. *Embedding* input sentences as word vectors;

[1] https://github.com/thunlp/TensorFlow-NRE

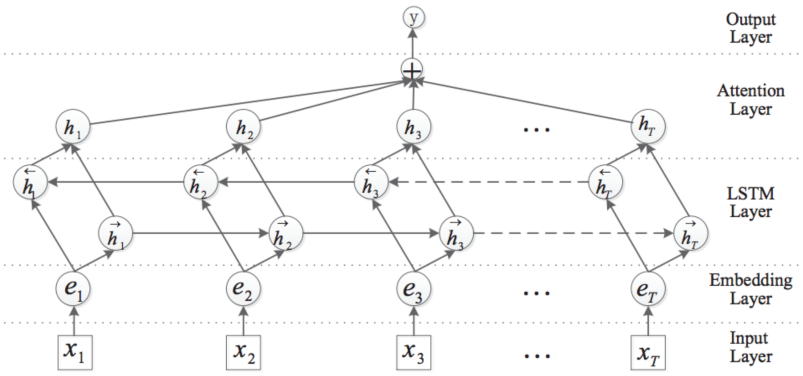


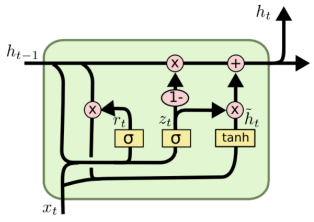
Figure 4: Bidirectional LSTM model with attention in (Zhou et al., 2016). In experiments, we used GRU cells instead.

2. *Bidirectional GRU cells* corresponding to the embedding layer. The number of GRU units in one layer, and the number of GRU layers are both hyper-parameterizable;
3. *Attention* layer that "soft align" GRU outputs to the target, by multiplying a weight vector.

Input Vector Representation There are many paradigms of word embedding; converting human-readable vocabularies into numbers that can be processed by machine. Two major approaches from the past several years are the frequency-based embedding and prediction-based embedding.

There are two embedding steps done for the input vector representation. One is word embedding, which aims to capture syntactic and semantic meanings of the words. The other one is based on the assumption that words that close to the target entities are usually informative to determine the relation between entities. In experiment, we have adopted pre-trained word vectors by GloVe (Pennington et al., 2014), a frequency-based language model doing dimension reduction of words' co-occurrence matrix by first normalizing global word counts, and log-smoothing.

Bidirectional GRU GRU is one of the variations of Recurrent Neural Networks, where in general, at each time t, each hidden state h_t will receive both the input vector at time t, x_t and also receives from the same hidden layer in the previous time step $h_t - 1$. Besides, GRU has "update gates" combining input and forget gates, and it merges cell and hidden state as in Figure 5. We choose GRU over LSTM, because GRU is less expensive than LSTM, yet has more parameters to train than conventional RNN. The GRU cell forms each component in the LSTM/GRU layer in Figure 4.



$$z_t = \sigma\left(W_z \cdot [h_{t-1}, x_t]\right)$$
$$r_t = \sigma\left(W_r \cdot [h_{t-1}, x_t]\right)$$
$$\tilde{h}_t = \tanh\left(W \cdot [r_t * h_{t-1}, x_t]\right)$$
$$h_t = (1 - z_t) * h_{t-1} + z_t * \tilde{h}_t$$

Figure 5: The Gated Recurrent Unit (GRU) for RNN(Olah, 2015).

With each GRU component unchanged, bidirectional networks extend a unidirectional network by a second sequence of GRU cells, where the hidden-to-hidden connections take input in the opposite order. Since both GRU and LSTM have forget gate that may lose long-term sequential inputs, and a unidirectional layer cannot foresee future inputs, the introduction of bidirection alleviates both concerns. Thus, with both forward and backward hidden outputs, the output of i^{th} word becomes:

$$h_i = [\overrightarrow{h_i} \oplus \overleftarrow{h_i}].$$

However, (Zhou et al., 2016) and the package we adapt use element-wise sum to combine the forward $\overrightarrow{h_i}$ and backward $\overleftarrow{h_i}$ pass, which contradicts with bidirection convention of concatenation, so we modify this when experimenting.

Cost Function In the final output layer, for a sentence S, the model uses softmax to predict the relation label from the set of all relation classes Y:

$$\hat{p}(y|S) = softmax(W^{(S)}h^* + b^{(S)}),\ \hat{y} = \arg\max_y \hat{p}(y|S).$$

The paper uses normalized cross entropy of predicted relation and groundtruth label as the cost function, plus norm-2 regularization:

$$J(\theta) = -\frac{1}{m}\sum_{i=1}^{m} t_i \log(y_i) + \lambda\|\theta\|^2,$$

where $\mathbf{t} \in \mathbb{R}^m$ is the one-hot ground truth label, and $\mathbf{y} \in \mathbb{R}^m$ is the estimated probability distribution of the softmax layer.

4 Experiments

4.1 Datasets

We used two groups of datasets for experiments:

1. For exploring model performances via regularization and tuning hyper-parameters: **NYT10** originally released by Riedel et al. (2010), and pre-trained word vectors from New York Times Annotated Corpus, which are also suggested in (Lin et al., 2016).
2. For further extracting more general relations, and using as sources for graphical models: **Wikipedia** Biography Dataset, and pre-trained GloVe word vectors from Wikipedia 2014 and Gigaword 5.

4.2 Evaluation Metric

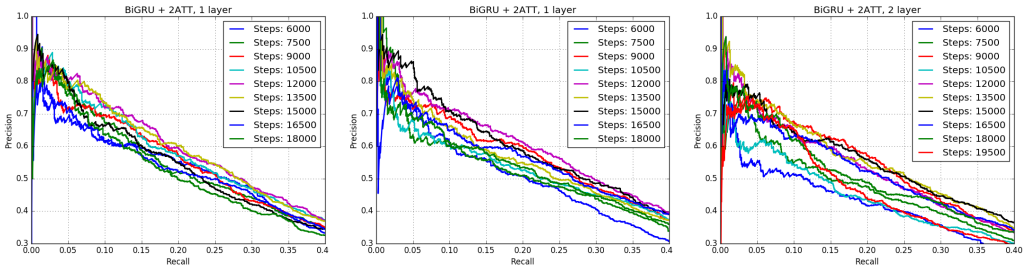
We will evaluate each model on precision, recall and area under curve to be consistent with the literature.

4.3 Variations of training tricks

The experiments under this section are conducted with NYT10 dataset, with pre-trained word vectors from NYTAC. We used mini-batch gradient descent in training, with fixed 3 epochs.

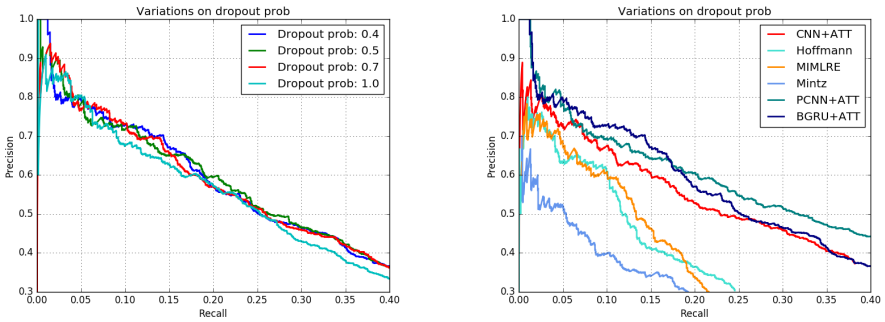
Bi-BRU concatenation & GRU hyper-parameters In most literatures, backward and forward output vectors are concatenated, while in the package, we found they're element-wisely summed up. We enlarged the bi-directional output vector dimension for attention, and followed to use concatenation instead. Besides, We have tuned some important hyper-parameters, *e.g.,* Adam optimizer learning rate, GRU unit size, number of layers, *etc.*, while the best performances are relatively stable towards distinct hyper-parameters. Figure 6 illustrates how the precision-recall curve evolves on the testing set from models with increasing training steps, *etc*. Comparing the three sub-graphs we can find out, for the models with same training step, different unit size of GRU has subtle effects on the performance, while 2-layer performs worse than 1-layer models in general. Most significantly from each sub-graph, number of training steps is the key factor towards the overall performance. Due to the small training size, it's more likely to overfit on the training data, rather than boost the performance via simple tuning.

Regularization and overfitting Previous tunings led us to explore regularization techniques. We used dropout on both forward and backward GRU cells with different keeping probabilities. Figure 7 shows the precision-recall curve for several different dropout keeping probabilities, with other hyper-parameters untouched and the same training step, from which we found out dropout regularization has non-trivial effect on the performance.



(a) 100-unit GRU (b) 230-unit GRU (c) 230-unit GRU

Figure 6: Testing results for BiGRU with attentions on the NYT dataset, for models with different training steps. a and b uses 1 layer network, and c uses 2 layers.



(a) Precision-recall curve for different dropout keeping probabilities with selective training step.

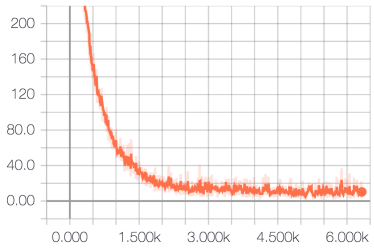
(b) BiGRU+ATT compared with other state-of-art models in literature, prob = 0.4.

Figure 7: Selective dropout comparison results.

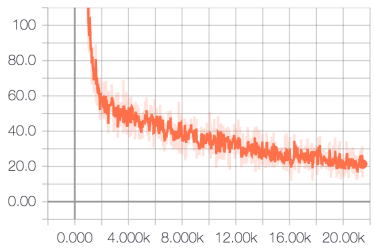
Then to quantitatively observe the relation between regularization and training steps, it's straight forward to calculate the area under the precision-recall curve, as in Table 1. By comparing the best performance (largest area under curve) of each probability, the less regularized (higher dropout probability, except for without dropout) models are easier to be overfitting/should have smaller training steps. However, their best performances are relatively close.

Table 1: Area under precision-recall curve for different dropout probabilities of increasing training steps.

Steps	Prob = 0.4	Prob = 0.5	Prob = 0.7	No dropout
6000	0.2748	0.2807	0.2820	0.2799
7000	0.3096	0.3098	0.3121	0.3093
8000	0.3107	0.3084	0.3092	0.3108
9000	0.2974	0.3111	**0.3237**	0.3167
10000	0.3070	0.2953	0.3065	0.3004
11000	0.3273	**0.3271**	0.3203	**0.3241**
12000	0.3210	0.3135	0.3171	0.3167
13000	0.3114	0.3128	0.3046	0.2987
14000	0.3276	0.3198	0.3183	0.3084
15000	**0.3313**	0.3191	0.3135	0.3172
16000	0.3135	0.2825	0.2892	0.3029
17000	0.3028	0.2938	0.2867	0.3124



(a) NYT10 dataset



(b) Wiki dataset

Figure 8: Training loss w.r.t. training steps. Outputs by `TensorBoard`.

4.4 Sample outputs

A good model can be general to different datasets. As mentioned in 4.1, we (1) did the previous experiments on NYT10 dataset, which contains well structured and preprocessed relations (entity pairs usually have strong/general inference relations), and also adopt Wiki dataset to our model, whose entities are sparse compared to relations. As expected, 8a shows training loss on NYT10 data converges quickly w.r.t training steps, while the loss on Wiki (8b) is significantly larger. However, precision-recall values on Wiki's *test* set performs better than NYT. This may also due to the sparsity of Wiki entities, that as along as the model has learnt information for the testing pair, model can more easily figure out the correct relation class, because the distribution of entity pairs are also sparse in the target classification space. Thus, in the preprocessing of Wiki data, we have only kept the top 100 occurred relations, and filtered out rare ones. Table 2 gives a sample output on both datasets.

Table 2: Sample extracted relations on testing set.

	`entity1`	`entity2`	relation
NYT10	norway	kongsberg	/location/location/contains
NYT10	faik_konica	albania	/people/person/nationality
Wiki	thomas_towie	forward	position
Wiki	alex_müller_racing_driver	german	nationality

5 Furtherwork

Both statistical and deep learning models we have discussed are in a supervised manner. In fact, many relations in the dataset are not given, which forces the model to ignore them. There will be huge improvement on inferencing these un-labeled relations from trained supervised model.

5.1 Reinforcement learning

Wenhan et al., 2017 have studied the problem of learning to reason in knowledge graphs (KGs) specifically in large scale. A novel reinforcement learning framework for learning multi-hop relational paths is introduced: A policy-based agent with continuous states based on knowledge graph embeddings, which reasons in a KG vector-space by sampling the most promising relation to extend its path. In contrast to prior work, such approach includes a reward function that takes the accuracy, diversity, and efficiency into consideration. Experimentally, results from this paper show that the proposed method outperforms a path-ranking based algorithm and knowledge graph embedding methods on Freebase and Never-Ending Language Learning datasets.

5.2 Global Constraints on Relations

Most of Deep Learning models only consider the task of predicting a single relationship from two entities. However, in practice we often like to predict the relationship of the subject with many entities. In the Deep Learning context, predicting each of these relations independently fails to capture the global context of a document. For example if an article is about a musician and we know that the "genre" relationship is present in the text, we would expect other relationships such as "instrument",

or "debut-date" to have a higher probability of showing in the text. One way in which we would model these constraints is to use a graphical model; such as Linear CRF, to take the output of the Deep Learning work to represent distributions over many random variables and model the joint distribution of relationships.

5.3 Multi-instance Single-label Problem

One of the drawbacks in this model is that the model only consider one sentence. However; in real life, we can barely express a concrete relationship in one or two sentences. In addition, the model assumes that the relation pair between two entities always has the same target label. Multi-instance multi-label convolutional neural network relaxes the expressed-at-least-once assumption given in (Riedel et al., 2010); which makes strong assumption that "if two entities participate in a relation, at least one sentence that mentions these two entities will express the relation". Also, to the opposite of feeding one sentence into the network, MIML deploys cross-sentence max-pooling for better information sharing. Implementation from (Jiang et al., 2016) has shown consistent better results compared to PCNN.

6 Conclusion

In this project, we experiment and investigate many current and past algorithms on Relation Extraction. We conclude that by devising more attention mechanism and bi-GRU with CNN Deep Learning, the results come out substantially better in both public and our own testing datasets. Distant Supervision and PCNN emphasize on providing more attention on the target entities. While BGRU adds extra strength in sentence-level information sharing. Our experiments have shown the results that having these features in the model will steadily improve precision and recall. We will explore the topics mentioned in further work to encode more complicated correlations between relation paths.

7 Resources and Modifications

Our modifications are mainly based on thunlp/TensorFlow-NRE:

- The package only supports for `TensorFlow` r0.11, so we upgraded the codes to support latest version;
- Modifications for experiments, e.g., bi-direction concatenation, tunings and regularization
- Added preprocessing codes for Wiki dataset, and changed some hard-coded settings in the package for a new dataset.

We have also used GloVe pre-trained vectors (Pennington et al., 2014). DavidGrangier/wikipedia-biography-dataset

References

Mintz, M.; Bills, S.; Snow, R.; Jurafsky, D. Distant Supervision for Relation Extraction Without Labeled Data. 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. Stroudsburg, PA, USA, 2009; pp 1003–1011.

Riedel, S.; Yao, L.; McCallum, A. Modeling Relations and Their Mentions without Labeled Text. Machine Learning and Knowledge Discovery in Databases: European Conference, ECML PKDD 2010, Barcelona, Spain, September 20-24, 2010, Proceedings, Part III. Berlin, Heidelberg, 2010; pp 148–163.

Bahdanau, D.; Cho, K.; Bengio, Y. Neural Machine Translation by Jointly Learning to Align and Translate. 2014.

Blunsom, P. Deep Natural Language Processing. 2017; `https://github.com/oxford-cs-deepnlp-2017/lectures`.

Zeng, D.; Liu, K.; Lai, S.; Zhou, G.; Zhao, J. Relation classification via convolutional deep neural network. the 25th International Conference on Computational Linguistics: Technical Papers. 2014; pp 2335–2344.

Zeng, D.; Liu, K.; Chen, Y.; Zhao, J. Distant Supervision for Relation Extraction via Piecewise Convolutional Neural Networks. Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing, EMNLP 2015, Lisbon, Portugal, September 17-21, 2015. 2015; pp 1753–1762.

Nguyen, T.; Grishman, R. *Workshop on Vector Space Modeling for NLP at NAACL 2015*; 2015.

Lin, Y.; Shen, S.; Liu, Z.; Luan, H.; Sun, M. Neural Relation Extraction with Selective Attention over Instances. Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics, ACL 2016, August 7-12, 2016, Berlin, Germany, Volume 1: Long Papers. 2016.

Zhang, S.; Zheng, D.; Hu, X.; Yang, M. Bidirectional Long Short-Term Memory Networks for Relation Classification. PACLIC. 2015.

Zhou, P.; Shi, W.; Tian, J.; Qi, Z.; Li, B.; Hao, H.; Xu, B. Attention-Based Bidirectional Long Short-Term Memory Networks for Relation Classification. ACL. 2016.

Pennington, J.; Socher, R.; Manning, C. D. GloVe: Global Vectors for Word Representation. Empirical Methods in Natural Language Processing (EMNLP). 2014; pp 1532–1543.

Olah, C. Understanding LSTM Networks. 2015; `http://colah.github.io/posts/2015-08-Understanding-LSTMs/`.

Riedel, S.; Yao, L.; McCallum, A. Modeling Relations and Their Mentions Without Labeled Text. Proceedings of the 2010 European Conference on Machine Learning and Knowledge Discovery in Databases: Part III. Berlin, Heidelberg, 2010; pp 148–163.

Jiang, X.; Wang, Q.; Li, P.; Wang, B. Relation Extraction with Multi-instance Multi-label Convolutional Neural Networks. COLING. 2016.