Jointly Extracting Relations with Class Ties via Effective Deep Ranking

Hai Ye[1], **Wenhan Chao**[1], **Zhunchen Luo**[2*], **Zhoujun Li**[1]
[1]School of Computer Science and Engineering, Beihang University, Beijing 100191, China
{yehai, chaowenhan, lizj}@buaa.edu.cn
[2]China Defense Science and Technology Information Center, Beijing 100142, China
zhunchenluo@gmail.com

Abstract

Connections between relations in relation extraction, which we call *class ties*, are common. In distantly supervised scenario, one entity tuple may have multiple relation facts. Exploiting class ties between relations of one entity tuple will be promising for distantly supervised relation extraction. However, previous models are not effective or ignore to model this property. In this work, to effectively leverage class ties, we propose to make joint relation extraction with a unified model that integrates convolutional neural network (CNN) with a general pairwise ranking framework, in which three novel ranking loss functions are introduced. Additionally, an effective method is presented to relieve the severe class imbalance problem from NR (not relation) for model training. Experiments on a widely used dataset show that leveraging class ties will enhance extraction and demonstrate the effectiveness of our model to learn class ties. Our model outperforms the baselines significantly, achieving state-of-the-art performance.

1 Introduction

Relation extraction (RE) aims to classify the relations between two given named entities from natural-language text. Supervised machine learning methods require numerous labeled data to work well. With the rapid growth of volume of relation types, traditional methods can not keep up with the step for the limitation of labeled data. In order to narrow down the gap of data sparsity, Mintz et al. (2009) propose *distant supervision (DS)* for relation extraction, which automatically generates training data by aligning a knowledge facts database (ie. Freebase (Bollacker et al., 2008)) with texts.

place_lived (Patsy Ramsey, Atlanta)
place_of_birth (Patsy Ramsey, Atlanta)

	Sentence	Latent Label
#1	*Patsy Ramsey* has been living in *Atlanta* since she was born.	*place_of_birth*
#2	*Patsy Ramsy* always loves *Atlanta* since it is her hometown.	*place_lived*

Table 1: Training instances generated by freebase.

Class ties mean the connections between relations in relation extraction. In general, we conclude that class ties can have two types: weak class ties and strong class ties. Weak class ties mainly involve the co-occurrence of relations such as *place_of_birth* and *place_lived*, *CEO_of* and *founder_of*. On the contrary, strong class ties mean that relations have latent logical entailments. Take the two relations of *capital_of* and *city_of* for example, if one entity tuple has the relation of *capital_of*, it must express the relation fact of *city_of*, because the two relations have the entailment of *capital_of* ⇒ *city_of*. Obviously the opposite induction is not correct. Further take the sentence of *"Jonbenet told me that her mother [Patsy Ramsey]*$_{e_1}$ *never left [Atlanta]*$_{e_2}$ *since she was born."* in DS scenario for example. This sentence expresses two relation facts which are *place_of_birth* and *place_lived*. However, the word "born" is a strong bios to extract *place_of_birth*, so it may not be easy to predict the relation of *place_lived*, but if we can incorporate the weak ties between the two relations, extracting *place_of_birth* will provide evidence for prediction of *place_lived*.

Exploiting class ties is necessary for DS based relation extraction. In DS scenario, there is a challenge that one entity tuple can have multiple rela-

[*] Corresponding author.

tion facts as shown in Table 1, which is called *relation overlapping* (Hoffmann et al., 2011; Surdeanu et al., 2012). However, the relations of one entity tuple can have class ties mentioned above which can be leveraged to enhance relation extraction for it narrowing down potential searching spaces and reducing uncertainties between relations when predicting unknown relations. If one pair entities has *CEO_of*, it will contain *founder_of* with high possibility.

To exploit class ties between relations, we propose to make joint extraction for all positive labels of one entity tuple with considering *pairwise* connections between positive and negative labels inspired by (Fürnkranz et al., 2008; Zhang and Zhou, 2006). As the two relations with class ties shown in Table 1, by joint extraction of two relations, we can maintain the *class ties* (co-occurrence) of them from training samples to be learned by potential model, and then leverage this learned information to extract instances with unknown relations, which can not be achieved by separated extraction for it dividing labels apart losing information of co-occurrence. To classify positive labels from negative ones, we adopt pairwise ranking to rank positive ones higher than negative ones, exploiting pairwise connections between them. In a word, joint extraction exploits class ties between relations and pairwise ranking classify positive labels from negative ones. Furthermore, combining information across sentences will be more appropriate for joint extraction which provides more information from other sentences to extract each relation (Zheng et al., 2016; Lin et al., 2016). In Table 1, sentence #1 is the evidence for *place_of_birth*, but it also expresses the meaning of "living in someplace", so it can be aggregated with sentence #2 to extract *place_lived*. Meanwhile, the word of "hometown" in sentence #2 can provide evidence for *place_of_birth* which should be combined with sentence #1 to extract *place_of_birth*.

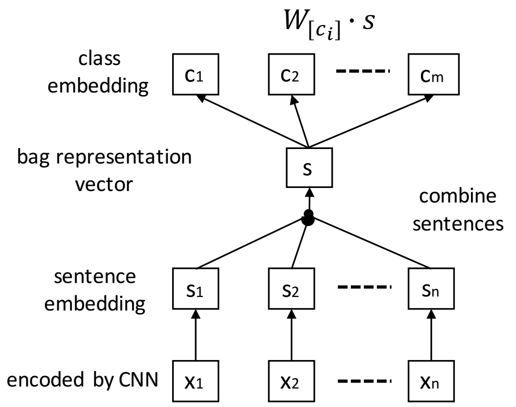

Figure 1: The main architecture of our model.

In this work, we propose a unified model that integrates pairwise ranking with CNN to exploit class ties. Inspired by the effectiveness of deep learning for modeling sentences (LeCun et al., 2015), we use CNN to encode sentences. Similar to (Santos et al., 2015; Lin et al., 2016), we use class embeddings to represent relation classes. The whole model architecture is presented in Figure 1. We first use CNN to embed sentences, then we introduce two variant methods to combine the embedded sentences into one bag representation vector aiming to aggregate information across sentences, after that we measure the similarity between bag representation and relation class in real-valued space. With two variants for combining sentences, three novel pairwise ranking loss functions are proposed to make joint extraction. Besides, to relieve the bad impact of class imbalance from NR (not relation) (Japkowicz and Stephen, 2002) for training our model, we cut down loss propagation from NR class during training.

Our experimental results on dataset of Riedel et al. (2010) are evident that: (1) Our model is much more effective than the baselines; (2) Leveraging class ties will enhance relation extraction and our model is efficient to learn class ties by joint extraction; (3) A much better model can be trained after relieving class imbalance from NR.

Our contributions in this paper can be encapsulated as follows:

- We propose to leverage class ties to enhance relation extraction. An effective deep ranking model which integrates CNN and pairwise ranking framework is introduced to exploit class ties.
- We propose an effective method to relieve the impact of data imbalance from NR for model training.
- Our method achieves state-of-the-art performance.

2 Related Work

We summarize related works on two main aspects:

2.1 Distant Supervision Relation Extraction

Previous works on DS based RE ignore or are not effective to leverage class ties between relations.

Riedel et al. (2010) introduce multi-instance learning to relieve the wrong labelling problem, ignoring class ties. Afterwards, Hoffmann et al. (2011) and Surdeanu et al. (2012) model this problem by multi-instance multi-label learning to extract overlapping relations. Though they also propose to make joint extraction of relations, they only use information from single sentence losing information from other sentences. Han and Sun (2016) try to use *Markov logic* model to capture consistency between relation labels, on the contrary, our model leverages deep ranking to learn class ties automatically.

With the remarkable success of deep learning in CV and NLP (LeCun et al., 2015), deep learning has been applied to relation extraction (Zeng et al., 2014, 2015; Santos et al., 2015; Lin et al., 2016), the specific deep learning architecture can be CNN (Zeng et al., 2014), RNN (Zhou et al., 2016), etc. Zeng et al. (2015) propose a piecewise convolutional neural network with multi-instance learning for DS based relation extraction, which improves the precision and recall significantly. Afterwards, Lin et al. (2016) introduce the mechanism of attention (Luong et al., 2015; Bahdanau et al., 2014) to select the sentences to relieve the wrong labelling problem and use all the information across sentences. However, the two deep learning based models only make separated extraction thus can not model class ties between relations.

2.2 Deep Learning to Rank

Deep learning to rank has been widely used in many problems to serve as a classification model. In image retrieval, Zhao et al. (2015) apply deep semantic ranking for multi-label image retrieval. In text matching, Severyn and Moschitti (2015) adopt learning to rank combined with deep CNN for short text pairs matching. In traditional supervised relation extraction, Santos et al. (2015) design a pairwise loss function based on CNN for single label relation extraction. Based on the advantage of deep learning to rank, we propose pairwise learning to rank (LTR) (Liu, 2009) combined with CNN in our model aiming to jointly extract multiple relations.

3 Proposed Model

In this section, we first conclude the notations used in this paper, then we introduce the used CNN for sentence embedding, afterwards, we present our algorithm of how to learn class ties between relations of one entity tuple.

3.1 Notation

We define the relation classes as $\mathcal{L} = \{1, 2, \cdots, C\}$, entity tuples as $\mathcal{T} = \{t_i\}_{i=1}^{M}$ and mentions[1] as $\mathcal{X} = \{x_i\}_{i=1}^{N}$. Dataset is constructed as follows: for entity tuple $t_i \in \mathcal{T}$ and its relation class set $L_i \subseteq \mathcal{L}$, we collect all the mentions X_i that contain t_i, the dataset we use is $\mathcal{D} = \{(t_i, L_i, X_i)\}_{i=1}^{H}$. Given a data $(t_k, L_k, X_k) \in \{(t_i, L_i, X_i)\}_{i=1}^{H}$, the sentence embeddings of X_k encoded by CNN are defined as $S_k = \{s_i\}_{i=1}^{|X_k|}$ and we use class embeddings $W \in R^{|\mathcal{L}| \times d}$ to represent the relation classes.

3.2 CNN for Sentence Embedding

We take the effective CNN architecture adopted from (Zeng et al., 2015; Lin et al., 2016) to encode sentence and we briefly introduce CNN in this section. More details of our CNN can be obtained from previous work.

3.2.1 Words Representations

- **Word Embedding** Given a word embedding matrix $V \in \mathbb{R}^{l^w \times d^1}$ where l^w is the size of word dictionary and d^1 is the dimension of word embedding, the words of a mention $x = \{w_1, w_2, \cdots, w_n\}$ will be represented by real-valued vectors from V.
- **Position Embedding** The position embedding of a word measures the distance from the word to entities in a mention. We add position embeddings into words representations by appending position embedding to word embedding for every word. Given a position embedding matrix $P \in \mathbb{R}^{l^p \times d^2}$ where l^p is the number of distances and d^2 is the dimension of position embeddings, the dimension of words representations becomes $d^w = d^1 + d^2 \times 2$.

3.2.2 Convolution, Piecewise max-pooling

After transforming words in x to real-valued vectors, we get the sentence $q \in \mathbb{R}^{n \times d^w}$. The set of kernels K is $\{K_i\}_{i=1}^{d^s}$ where d^s is the number of kernels. Define the window size as d^{win} and given one kernel $K_k \in \mathbb{R}^{d^{win} \times d^w}$, the convolution operation is defined as follows:

$$m_{[i]} = q_{[i:i+d^{win}-1]} \odot K_k + b_{[k]} \qquad (1)$$

[1] The sentence containing one certain entity is called mention.

where m is the vector after conducting convolution along q for $n - d^{win} + 1$ times and $b \in \mathbb{R}^{d^s}$ is the bias vector. For these vectors whose indexes out of range of $[1, n]$, we replace them with zero vectors.

By piecewise max-pooling, when pooling, the sentence is divided into three parts: $m_{[p_0:p_1]}$, $m_{[p_1:p_2]}$ and $m_{[p_2:p_3]}$ (p_1 and p_2 are the positions of entities, p_0 is the beginning of sentence and p_3 is the end of sentence). This piecewise max-pooling is defined as follows:

$$z_{[j]} = max(m_{[p_{j-1}:p_j]}) \tag{2}$$

where $z \in \mathbb{R}^3$ is the result of mention x processed by kernel K_k; $1 \leq j \leq 3$. Given the set of kernels K, following the above steps, the mention x can be embedded to o where $o \in R^{d^s * 3}$.

3.2.3 Non-Linear Layer, Regularization

To learn high-level features of mentions, we apply a non-linear layer after pooling layer. After that, a dropout layer is applied to prevent overfitting. We define the final fixed sentence representation as $s \in \mathbb{R}^{d^f}$ ($d^f = d^s * 3$).

$$s = g(o) \circ h \tag{3}$$

where $g(\cdot)$ is a non-linear function and we use $tanh(\cdot)$ in this paper; h is a Bernoulli random vector with probability p to be 1.

3.3 Learning Class Ties by Joint Extraction with Pairwise Ranking

As mentioned above, to learn class ties, we propose to make joint extraction with considering pairwise connections between positive labels and negative ones. Pairwise ranking is applied to achieve this goal. Besides, combining information across sentences is necessary for joint extraction. More specifically, as shown in Figure 2, from down to top, all information from sentences is pre-propagated to provide enough information for joint extraction. From top to down, pairwise ranking jointly extracting positive relations by combining losses, which are back-propagated to CNN to learn class ties.

3.3.1 Combining Information across Sentences

We propose two options to combine sentences to provide enough information for joint extraction.

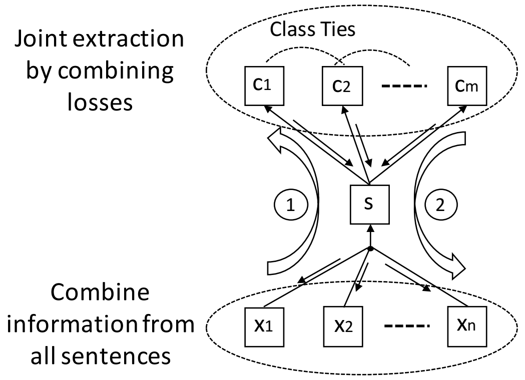
Joint extraction by combining losses
Class Ties
C1
C2
Cm
1
S
2
Combine information from all sentences
X1
X2
Xn

Figure 2: Illustration of mechanism of our model to model class ties between relations.

- **AVE** The first option is average method. This method regards all the sentences equally and directly average the values in all dimensions of sentence embedding. This **AVE** function is defined as follows:

$$s = \frac{1}{n} \sum_{s_i \in S_k} s_i \tag{4}$$

where n is the number of sentences and s is the representation vector combining all sentence embeddings. Because it weights the importance of sentences equally, this method may bring much noise data from two aspects: (1) the wrong labelling data; (2) irrelated mentions for one relation class, for all sentences containing the same entity tuple being combined together to construct the bag representation.

- **ATT** The second one is a sentence-level attention algorithm used by Lin et al. (2016) to measure the importance of sentences aiming to relieve the wrong labelling problem. For every sentence, **ATT** will calculate a weight by comparing the sentence to one relation. We first calculate the similarity between one sentence embedding and relation class as follows:

$$e_j = a \cdot W_{[c]} \cdot s_j \tag{5}$$

where e_j is the similarity between sentence embedding s_j and relation class c and a is a bias factor. In this paper, we set a as 0.5. Then we apply Softmax to rescale $\mathbf{e}$ ($\mathbf{e} = \{e_i\}_{i=1}^{|X_k|}$) to $[0, 1]$. We get the weight α_j for s_j as follows:

$$\alpha_j = \frac{\exp(e_j)}{\sum_{e_i \in \mathbf{e}} \exp(e_i)} \tag{6}$$

so the function to merge s with **ATT** is as follows:

$$s = \sum_{i=1}^{|X_k|} \alpha_i \cdot s_i \tag{7}$$

3.3.2 Joint Extraction by Combining Losses to Learn Class Ties

Firstly, we have to present the score function to measure the similarity between s and relation c.

• **Score Function** We use dot function to produce score for s to be predicted as relation c. The score function is as follows:

$$\mathcal{F}(s, c) = W_{[c]} \cdot s \tag{8}$$

There are other options for score function. In Wang et al. (2016), they propose a margin based loss function that measures the similarity between s and $W_{[c]}$ by distance. Because score function is not an important issue in our model, we adopt dot function, also used by Santos et al. (2015) and Lin et al. (2016), as our score function.

Now we start to introduce the ranking loss function.

Pairwise ranking aims to learn the score function $\mathcal{F}(s, c)$ that ranks positive classes higher than negative ones. This goal can be summarized as follows:

$$\forall c^+ \in L_k, \forall c^- \in \mathcal{L}-L_k : \mathcal{F}(s, c^+) > \mathcal{F}(s, c^-)+\beta \tag{9}$$

where β is a margin factor which controls the minimum margin between the positive scores and negative scores.

To learn class ties between relations, we extend the formula (9) to make joint extraction and we propose three ranking loss functions with variants of combining sentences. Followings are the proposed loss functions:

• **with AVE (Variant-1)** We define the margin-based loss function with option of **AVE** to aggregate sentences as follows:

$$\begin{aligned} G_{[ave]} = \sum_{c^+ \in L_k} & \rho[0, \sigma^+ - \mathcal{F}(s, c^+)]_+ \\ & +\rho|L_k|[0, \sigma^- + \mathcal{F}(s, c^-)]_+ \end{aligned} \tag{10}$$

where $[0, \cdot\,]_+ = max(0, \cdot\,)$; ρ is the rescale factor, σ^+ is positive margin and σ^- is negative margin. Similar to Santos et al. (2015) and Wang et al. (2016), this loss function is designed to rank positive classes higher than negative ones controlled by the margin of $\sigma^+ - \sigma^-$. In reality, $\mathcal{F}(s, c^+)$ will be higher than σ^+ and $\mathcal{F}(s, c^-)$ will be lower than σ^-. In our work, we set ρ as 2, σ^+ as 2.5 and σ^- as 0.5 adopted from Santos et al. (2015).

Similar to Weston et al. (2011) and Santos et al. (2015), we update one negative class at every training round but to balance the loss between positive classes and negative ones, we multiply $|L_k|$ before the right term in function (10) to expand the negative loss. We apply mini-bach based stochastic gradient descent (SGD) to minimize the loss function. The negative class is chosen as the one with highest score among all negative classes (Santos et al., 2015), i.e.:

$$c^- = \underset{c \in \mathcal{L}-L_k}{\operatorname{argmax}} \mathcal{F}(s, c) \tag{11}$$

• **with ATT (Variant-2)** Now we define the loss function for the option of **ATT** to combine sentences as follows:

$$\begin{aligned} G_{[att]} = \sum_{c^+ \in L_k} & (\rho[0, \sigma^+ - \mathcal{F}(s^{c^+}, c^+)]_+ \\ & +\rho[0, \sigma^- + \mathcal{F}(s^{c^+}, c^-)]_+) \end{aligned} \tag{12}$$

where s^c means the attention weights of representation s are merged by comparing sentence embeddings with relation class c and c^- is chosen by the following function:

$$c^- = \underset{c \in \mathcal{L}-L_k}{\operatorname{argmax}} \mathcal{F}(s^{c^+}, c) \tag{13}$$

which means we update one negative class in every training round. We keep the values of ρ, σ^+ and σ^- same as values in function (10).

According to this loss function, we can see that: for each class $c^+ \in L_k$, it will capture the most related information from sentences to merge s^{c^+}, then rank $\mathcal{F}(s^{c^+}, c^+)$ higher than all negative scores which each is $\mathcal{F}(s^{c^+}, c^-)$ $(c^- \in \mathcal{L} - L_k)$. We use the same update algorithm to minimize this loss.

• **Extended with ATT (Variant-3)** According to function (12), for each c^+, we only select one negative class to update the parameters, which only considers the connections between positive classes and negative ones, ignoring connections between positive classes, so we extend function (12) to better exploit class ties by considering the connections between positive classes. We give out the extended loss function as follows:

$$\begin{aligned} G_{[Exatt]} = \sum_{c^* \in L_k} (\sum_{c^+ \in L_k} & \rho[0, \sigma^+ - \mathcal{F}(s^{c^*}, c^+)]_+ \\ & +\rho[0, \sigma^- + \mathcal{F}(s^{c^*}, c^-)]_+) \end{aligned} \tag{14}$$

Pro.	Training	Test
SemE.	17.63%	16.71%
Riedel	72.52%	96.26%

Table 2: The proportions of NR samples from SemEval-2010 Task 8 dataset and Riedel dataset.

Similar to function (13), we select c^- as follows:

$$c^- = \operatorname*{argmax}_{c \in \mathcal{L} - L_k} \mathcal{F}(s^{c^*}, c) \tag{15}$$

and we use the same method to update this loss function as discussed above. From the function (14), we can see that: for $c^* \in L_k$, after merging the bag representation s with c^*, we share s with all the other positive classes and update the class embeddings of other positive classes with s, in this way, the connections between positive classes can be captured and learned by our model.

In loss function (10), (12) and (14), we combine losses from all positive labels to make joint extraction to capture the class ties among relations. Suppose we make separated extraction, the losses from positive labels will be divided apart and will not get enough information of connections between positive labels, comparing to joint extraction. Connections between positive labels and negative ones are exploited by controlling margins: σ^+ and σ^-.

3.4 Relieving Impact of NR

In relation extraction, the dataset will always contain certain negative samples which do not express relations classified as NR (not relation). Table 2 presents the proportion of NR samples in SemEval-2010 Task 8 dataset[2] (Erk and Strapparava, 2010) and dataset from Riedel et al. (2010), which shows almost data is about NR in the latter dataset. Data imbalance will severely affect the model training and cause the model only sensitive to classes with high proportion (He and Garcia, 2009).

In order to relieve the impact of NR in DS based relation extraction, we cut the propagation of loss from NR, which means if relation c is NR, we set its loss as 0. Our method is similar to Santos et al. (2015) with slight variance. Santos et al. (2015) directly omit the NR class embedding, but we keep it. If we use **ATT** method to combine information across sentences, we can not omit NR class embedding according to function (6) and (7), on the contrary, it will be updated from the negative classes' loss.

Algorithm 1: Merging loss function of **Variant-3**

```
input : $\mathcal{L}$, $(t_k, L_k, X_k)$ and $S_k$;
output: $G_{[Exatt]}$;
$G_{[Exatt]} \leftarrow 0$;
for $c^* \in L_k$ do
    Merge representation $s^{c^*}$ by function (5), (6), (7);
    for $c^+ \in L_k$ do
        if $c^+$ is not NR then
            $G_{[Exatt]} \leftarrow G_{[Exatt]} + \rho[0, \sigma^+ - \mathcal{F}(s^{c^*}, c^+)]_+$;
    $c^- \leftarrow \text{argmax}_{c \in \mathcal{L} - L_k} \mathcal{F}(s^{c^*}, c)$;
    $G_{[Exatt]} \leftarrow G_{[Exatt]} + \rho[0, \sigma^- + \mathcal{F}(s^{c^*}, c^-)]_+$;
return $G_{[Exatt]}$;
```

In Algorithm 1, we give out the pseudocodes of merging loss with **Variant-3** and considering to relieve the impact of NR.

4 Experiments

4.1 Dataset and Evaluation Criteria

We conduct our experiments on a widely used dataset, developed by Riedel et al. (2010) and has been used by Hoffmann et al. (2011), Surdeanu et al. (2012), Zeng et al. (2015) and Lin et al. (2016). The dataset aligns Freebase relation facts with the New York Times corpus, in which training mentions are from 2005-2006 corpus and test mentions from 2007.

Following Mintz et al. (2009), we adopt held-out evaluation framework in all experiments. Aggregated precision/recall curves are drawn and precision@N (P@N) is reported to illustrate the model performance.

4.2 Experimental Settings

Word Embeddings. We use a word2vec tool that is gensim[3] to train word embeddings on NYT corpus. Similar to Lin et al. (2016), we keep the words that appear more than 100 times to construct word dictionary and use "UNK" to represent the other ones.

[2] This is a dataset for relation extraction in traditional supervision framework.

[3] http://radimrehurek.com/gensim/models/word2vec.html

Parameter Name	Symbol	Value
Window size	d^{win}	3
Sentence. emb. dim.	d^f	690
Word. emb. dim.	d^1	50
Position. emb. dim.	d^2	5
Batch size	$\mathcal{B}$	160
Learning rate	λ	0.03
Dropout pos.	p	0.5

Table 3: Hyper-parameter settings.

Hyper-parameter Settings. Three-fold validation on the training dataset is adopted to tune the parameters following Surdeanu et al. (2012). We use grid search to determine the optimal hyper-parameters. We select word embedding size from $\{50, 100, 150, 200, 250, 300\}$. Batch size is tuned from $\{80, 160, 320, 640\}$. We determine learning rate among $\{0.01, 0.02, 0.03, 0.04\}$. The window size of convolution is tuned from $\{1, 3, 5\}$. We keep other hyper-parameters same as Zeng et al. (2015): the number of kernels is 230, position embedding size is 5 and dropout rate is 0.5. Table 3 shows the detailed parameter settings.

4.3 Comparisons with Baselines

Baseline. We compare our model with the following baselines:

- **Mintz** (Mintz et al., 2009) the original distantly supervised model.
- **MultiR** (Hoffmann et al., 2011) a multi-instance learning based graphical model which aims to address overlapping relation problem.
- **MIML** (Surdeanu et al., 2012) also solving overlapping relations in a multi-instance multi-label framework.
- **PCNN+ATT** (Lin et al., 2016) the state-of-the-art model in dataset of Riedel et al. (2010) which applies **ATT** to combine the sentences.

Results and Discussion. We compare our three variants of loss functions with the baselines and the results are shown in Figure 3. From the results we can see that: (1) Rank + AVE (Variant-1) achieves comparable results with PCNN+ATT; (2) Rank + ATT (Variant-2) and Rank + ExATT (Variant-3) significantly outperform PCNN + ATT with much higher precision and slightly higher recall in whole view; (3) Rank + ExATT (Variant-3) exhibits the best performances comparing with all the other methods including PCNN + ATT, Rank + AVE and Rank + ATT.

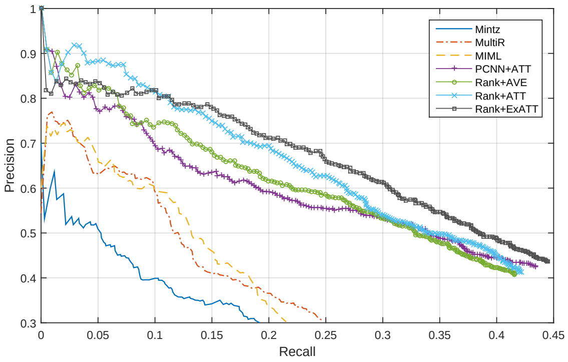

Figure 3: Performance comparison of our model and the baselines.

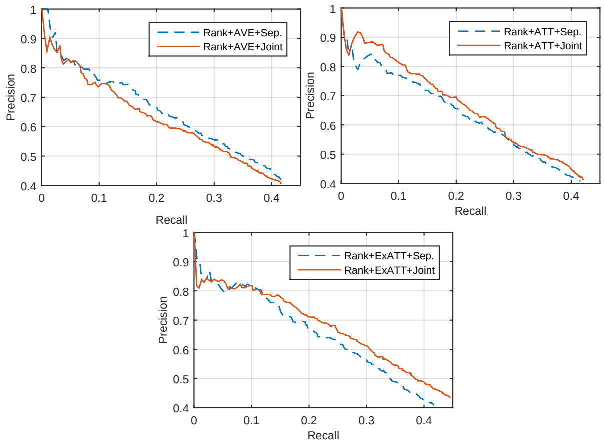

Figure 4: Results for impact of joint extraction and class ties with methods of Rank + AVE, Rank + ATT and Rank + ExATT under the setting of relieving impact of NR.

4.4 Impact of Joint Extraction and Class Ties

In this section, we conduct experiments to reveal the effectiveness of our model to learn class ties with three variant loss functions mentioned above, and the impact of class ties for relation extraction. As mentioned above, we make joint extraction to learn class ties, so to achieve the goal of this set of experiments, we compare joint extraction with separated extraction. To make separated extraction, we divide the labels of entity tuple into single label and for one relation label we only select the sentences expressing this relation, then we use this dataset to train our model with the three variant loss functions. We conduct experiments with Rank + AVE (Variant-1), Rank + ATT (Variant-2) and Rank + ExATT (Variant-3) relieving impact of NR. Aggregated P/R curves are drawn and precisions@N (100, 200, $\cdots$, 500) are reported to show the model performances.

P@N(%)	100	200	300	400	500	Ave.
R.+AVE+J.	81.3	76.4	74.6	69.6	66.0	73.6
R.+AVE+S.	**82.4**	**79.6**	74.6	**74.4**	**69.9**	**76.2**
R.+ATT+J.	**87.9**	**84.3**	**78.0**	**74.9**	**70.3**	**79.1**
R.+ATT+S.	82.4	79.1	75.9	71.9	69.5	75.7
R.+ExATT+J.	**83.5**	82.2	78.7	**77.2**	**73.1**	**79.0**
R.+ExATT+S.	82.4	**82.7**	**79.4**	74.2	69.2	77.6

Table 4: Precisions for top 100, 200, 300, 400, 500 and average of them for impact of joint extraction and class ties.

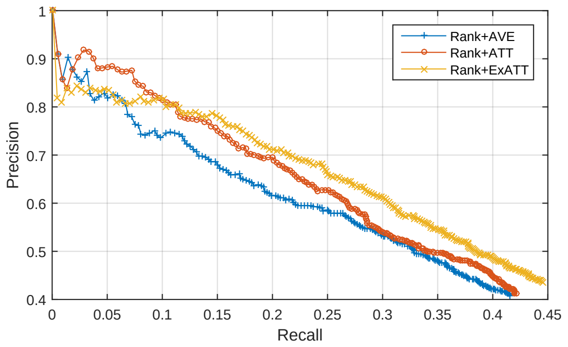

Figure 5: Results for comparisons of variant joint extractions.

P@N(%)	100	200	300	400	500	Ave.
R.+AVE	81.3	76.4	74.6	69.6	66.0	73.6
R.+ATT	**87.9**	**84.3**	78.0	74.9	70.3	**79.1**
R.+ExATT	83.5	82.2	**78.7**	**77.2**	**73.1**	79.0

Table 5: Precisions for top 100, 200, 300, 400, 500 and average of them for Rank + AVE, Rank + ATT and Rank + ExATT.

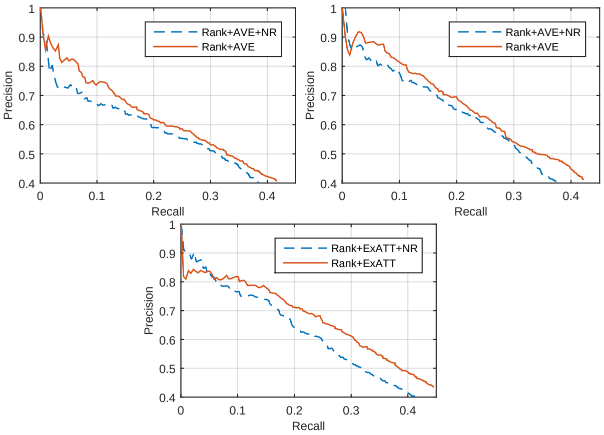

Figure 6: Results for impact of relation NR with methods of Rank + AVE, Rank + ATT and Rank + ExATT. "+NR" means not relieving impact of NR.

Experimental results are shown in Figure 4 and Table 4. From the results we can see that: (1) For Rank + ATT and Rank + ExATT, joint extraction exhibits better performance than separated extraction, which demonstrates class ties will improve relation extraction and the two methods are effective to learn class ties; (2) For Rank + AVE, surprisingly joint extraction does not keep up with separated extraction. For the second phenomenon, the explanation may lie in the **AVE** method to aggregate sentences will incorporate noise data consistent with the finding in Lin et al. (2016). When make joint extraction, we will combine all sentences containing the same entity tuple no matter which class type is expressed, so it will engender much noise if we only combine them equally.

4.5 Comparisons of Variant Joint Extractions

To make joint extraction, we have proposed three variant loss functions including Rank + AVE, Rank + ATT and Rank + ExATT in the above discussion and Figure 3 shows that the three variants achieve different performances. In this experiment, we aim to compare the three variants in detail. We conduct the experiments with the three variants under the setting of relieving impact of NR and joint extraction. We draw the P/R curves and report the top N (100, 200, $\cdots$, 500) precisions to compare model performance with the three variants.

From the results as shown in Figure 5 and Table 5 we can see that: (1) Comparing Rank + AVE with Rank + ATT, from the whole view, they can achieve the similar maximal recall point, but Rank + ATT exhibits higher precision in all range of recall; (2) Comparing Rank + ATT with Rank + ExATT, Rank + ExATT achieves much better performance with broader range of recall and higher precision in almost range of recall.

4.6 Impact of NR Relation

The goal of this experiment is to inspect how much relation of NR can affect the model performance. We use Rank + AVE, Rank + ATT, Rank + ExATT under the setting of relieving impact of NR or not to conduct experiments. We draw the aggregated P/R curves as shown in Figure 6, from which we can see that after relieving the impact of NR, the model performance can be improved significantly.

Then we further evaluate the impact of NR for convergence behavior of our model in model train-

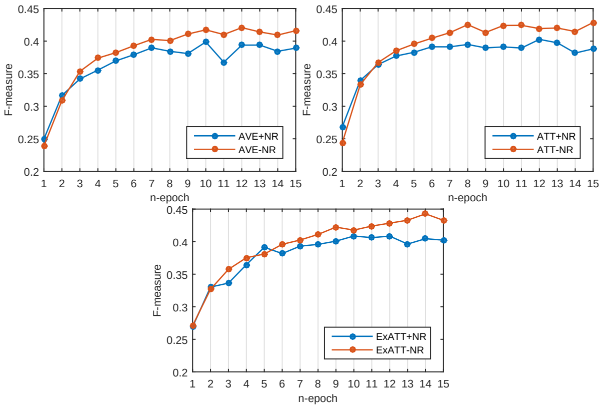

Figure 7: Impact of NR for model convergence. “+NR” means not relieving NR impact; “-NR” is opposite.

ing. Also with the three variant loss functions, in each iteration, we record the maximal value of F-measure [4] to represent the model performance at current epoch. Model parameters are tuned for 15 times and the convergence curves are shown in Figure 7. From the result, we can find out: “+NR” converges quicker than “-NR” and arrives to the final score at the around 11 or 12 epoch. In general, “-NR” converges more smoothly and will achieve better performance than “+NR” in the end.

4.7 Case Study

Joint vs. Sep. Extraction (Class Ties). We randomly select an entity tuple *(Cuyahoga County, Cleveland)* from test set to see its scores for every relation class with the method of Rank + ATT under the setting of relieving impact of NR with joint extraction and separated extraction. This entity tuple have two relations: */location/./county_seat* and */location/./contains*, which derive from the same root class and they have weak class ties for they all relating to topic of “location”. We rescale the scores by adding value 10. The results are shown in Figure 8, from which we can see that: under joint extraction setting, the two gold relations have the highest scores among the other relations but under separated extraction setting, only */location/./contains* can be distinguished from the negative relations, which demonstrates that joint extraction is better than separated extraction by capturing the class ties between relations.

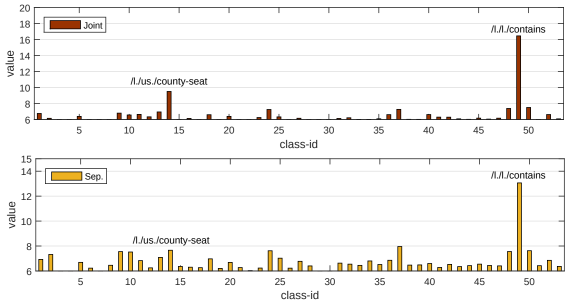

Figure 8: The output scores for every relation with method of Rank + ATT. The top is under joint extraction setting; the bottom is under separated extraction.

5 Conclusion and Future Works

In this paper, we leverage class ties to enhance relation extraction by joint extraction using pairwise ranking combined with CNN. An effective method is proposed to relieve the impact of NR for model training. Experimental results on a widely used dataset show that leveraging class ties will enhance relation extraction and our model is effective to learn class ties. Our method significantly outperforms the baselines.

In the future, we will focus on two aspects: (1) Our method in this paper considers pairwise intersections between labels, so to better exploit class ties, we will extend our method to exploit all other labels’ influences on each relation for relation extraction, transferring *second-order* to *high-order* (Zhang and Zhou, 2014); (2) We will focus on other problems by leveraging class ties between labels, specially on multi-label learning problems (Zhou et al., 2012) such as multi-category text categorization (Rousu et al., 2005) and multi-label image categorization (Zha et al., 2008).

Acknowledgments

Firstly, we would like to thank Xianpei Han and Kang Liu for their valuable suggestions on the initial version of this paper, which have helped a lot to improve the paper. Secondly, we also want to express gratitudes to the anonymous reviewers for their hard work and kind comments, which will further improve our work in the future. This work was supported by the National High-tech Research and Development Program (863 Program) (No. 2014AA015105) and National Natural Science Foundation of China (No. 61602490).

[4] $F = 2 * F * R/(P + R)$

References

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2014. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473* .

Kurt D. Bollacker, Colin Evans, Praveen Paritosh, Tim Sturge, and Jamie Taylor. 2008. Freebase: a collaboratively created graph database for structuring human knowledge. In *Proceedings of KDD*. pages 1247–1250.

Katrin Erk and Carlo Strapparava, editors. 2010. *Proceedings of SemEval*. The Association for Computer Linguistics.

Johannes Fürnkranz, Eyke Hüllermeier, Eneldo Loza Mencía, and Klaus Brinker. 2008. Multilabel classification via calibrated label ranking. *Machine learning* 73(2):133–153.

Xianpei Han and Le Sun. 2016. Global distant supervision for relation extraction. In *Proceedings of AAAI*. pages 2950–2956.

Haibo He and Edwardo A. Garcia. 2009. Learning from imbalanced data. *IEEE Trans. Knowl. Data Eng.* 21(9):1263–1284.

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 ACL-HLT*. Association for Computational Linguistics, pages 541–550.

Nathalie Japkowicz and Shaju Stephen. 2002. The class imbalance problem: A systematic study. *Intelligent data analysis* 6(5):429–449.

Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. 2015. Deep learning. *Nature* 521(7553):436–444.

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.

Tie-Yan Liu. 2009. Learning to rank for information retrieval. *Foundations and Trends in Information Retrieval* 3(3):225–331.

Thang Luong, Hieu Pham, and Christopher D. Manning. 2015. Effective approaches to attention-based neural machine translation. In *Proceedings of EMNLP*. pages 1412–1421.

Mike Mintz, Steven Bills, Rion Snow, and Dan Jurafsky. 2009. Distant supervision for relation extraction without labeled data. In *Proceedings of ACL-IJCNLP*. Association for Computational Linguistics, pages 1003–1011.

Sebastian Riedel, Limin Yao, and Andrew McCallum. 2010. Modeling relations and their mentions without labeled text. In *Proceedings of ECML-PKDD*. Springer, pages 148–163.

Juho Rousu, Craig Saunders, Sandor Szedmak, and John Shawe-Taylor. 2005. Learning hierarchical multi-category text classification models. In *Proceeding of ICML*. ACM, pages 744–751.

Cicero Nogueira dos Santos, Bing Xiang, and Bowen Zhou. 2015. Classifying relations by ranking with convolutional neural networks. In *Proceeding of ACL*. pages 626–634.

Aliaksei Severyn and Alessandro Moschitti. 2015. Learning to rank short text pairs with convolutional deep neural networks. In *Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval*. ACM, pages 373–382.

Mihai Surdeanu, Julie Tibshirani, Ramesh Nallapati, and Christopher D Manning. 2012. Multi-instance multi-label learning for relation extraction. In *Proceedings of EMNLP*. Association for Computational Linguistics, pages 455–465.

Linlin Wang, Zhu Cao, Gerard de Melo, and Zhiyuan Liu. 2016. Relation classification via multi-level attention cnns. In *Proceedings of ACL, Volume 1: Long Papers*.

Jason Weston, Samy Bengio, and Nicolas Usunier. 2011. WSABIE: scaling up to large vocabulary image annotation. In *Proceedings of IJCAI*. pages 2764–2770.

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 17–21.

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

Zheng-Jun Zha, Xian-Sheng Hua, Tao Mei, Jingdong Wang, Guo-Jun Qi, and Zengfu Wang. 2008. Joint multi-label multi-instance learning for image classification. In *CVPR*. IEEE, pages 1–8.

Min-Ling Zhang and Zhi-Hua Zhou. 2006. Multilabel neural networks with applications to functional genomics and text categorization. *IEEE transactions on Knowledge and Data Engineering* 18(10):1338–1351.

Min-Ling Zhang and Zhi-Hua Zhou. 2014. A review on multi-label learning algorithms. *IEEE transactions on knowledge and data engineering* 26(8):1819–1837.

Fang Zhao, Yongzhen Huang, Liang Wang, and Tieniu Tan. 2015. Deep semantic ranking based hashing for multi-label image retrieval. In *Proceedings of CVPR*. pages 1556–1564.

Hao Zheng, Zhoujun Li, Senzhang Wang, Zhao Yan, and Jianshe Zhou. 2016. Aggregating inter-sentence information to enhance relation extraction. In *Thirtieth AAAI Conference on Artificial Intelligence*.

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 *Proceeding of ACL*. page 207.

Zhi-Hua Zhou, Min-Ling Zhang, Sheng-Jun Huang, and Yu-Feng Li. 2012. Multi-instance multi-label learning. *Artificial Intelligence* 176(1):2291–2320.