Relation Extraction: Perspective from Convolutional Neural Networks

Thien Huu Nguyen
Computer Science Department
New York University
New York, NY 10003 USA
thien@cs.nyu.edu

Ralph Grishman
Computer Science Department
New York University
New York, NY 10003 USA
grishman@cs.nyu.edu

Abstract

Up to now, relation extraction systems have made extensive use of features generated by linguistic analysis modules. Errors in these features lead to errors of relation detection and classification. In this work, we depart from these traditional approaches with complicated feature engineering by introducing a convolutional neural network for relation extraction that automatically learns features from sentences and minimizes the dependence on external toolkits and resources. Our model takes advantages of multiple window sizes for filters and pre-trained word embeddings as an initializer on a non-static architecture to improve the performance. We emphasize the relation extraction problem with an unbalanced corpus. The experimental results show that our system significantly outperforms not only the best baseline systems for relation extraction but also the state-of-the-art systems for relation classification.

1 Introduction

Learning to extract semantic relations between entity pairs from text plays a vital role in information extraction, knowledge base population and question answering, to name a few. The *relation extraction* (RE) task can be divided into two steps: detecting if a relation utterance corresponding to some entity mention pair of interest in the same sentence represents some relation and classifying the detected relation mentions into some predefined classes. If we only need to categorize the given relation mentions that are known to express some expected relation (perfect detection), we are left with the *relation classification* (RC) task. One variation of relation classification is that one might have non-relation examples in the dataset but the number of those is comparable to the number of the other examples. The non-relation examples, therefore, can be treated as a usual relation class. Relation extraction, on the other hand, often comes with a tremendously unbalanced dataset where the number of the non-relation examples far exceeds the others, making relation extraction more challenging but more practical than relation classification. Our present work focuses on the relation extraction task with an unbalanced corpus.

In the last decade, the relation extraction literature has been dominated by two methods, distinguished by the nature of the relation representation: the feature-based method (Kambhatla, 2004; Boschee et al., 2005; Zhou et al., 2005; Grishman et al., 2005; Jiang and Zhai, 2007; Chan and Roth, 2010; Sun et al., 2011; Nguyen and Grishman, 2014) and the kernel-based method (Zelenko et al., 2003; Culotta and Sorensen, 2004; Bunescu and Mooney, 2005a; Bunescu and Mooney, 2005b; Zhang et al., 2006; Zhou et al., 2007; Qian et al., 2008; Nguyen et al., 2009; Sun and Han, 2014). The common characteristic of these methods is the leverage of a large body of linguistic analysis and knowledge resources to transform relation mentions into some rich representation to be used by some statistical classifier such as Support Vector Machines (SVM) or Maximum Entropy (MaxEnt). The linguistic analysis pipeline which is *hand-designed* itself includes tokenization, part of speech tagging, chunking, name tagging as well as parsing, often performed by existing natural language processing (NLP) modules.

Proceedings of NAACL-HLT 2015, pages 39–48,
Denver, Colorado, May 31 – June 5, 2015.

While these methods allow the RE systems to inherit the knowledge discovered by the NLP community for the pre-processing tasks, they might be subject to the error propagation introduced by the imperfect quality of the supervised NLP toolkits. For instance, all the tasks mentioned in the pipeline above are known to suffer from a performance loss when they are applied to out-of-domain data (Blitzer et al., 2006; Daumé III, 2007; McClosky et al., 2010), causing the collapse of the RE systems based on them. In this paper, we target an independent RE system that both avoids complicated feature engineering and minimizes the reliance on the supervised NLP modules for features, potentially alleviating the error propagation and advancing our performance in this area.

To be concrete, our relation extraction system is provided only with raw sentences marked with the positions of the two entities of interest[1]. The only elements we can derive from this structure are the words, the n-grams and their positions in the sentences, suggesting a paradigm in which relation mentions are represented by features that depend on these elements. Eventually, word embeddings that are capable of capturing latent semantic and syntactic properties of words (Bengio et al., 2001; Mnih and Hinton, 2007; Collobert and Weston, 2008; Mnih and Hinton, 2009; Turian et al., 2010; Mikolov et al., 2013) and convolutional neural networks (CNNs) that are able to recognize specific classes of n-gram and induce more abstract representations (Kalchbrenner et al., 2014) are a natural combination one should apply to obtain more effective representations for RE in this setting.

Convolutional neural networks (dating back to the 1980s) are a type of feed-forward artificial neural networks whose layers are formed by a convolution operation followed by a pooling operation (LeCun et al., 1988; Kalchbrenner et al., 2014). Recently, with the emerging interests of the community in deep learning, CNNs have been revived and effectively applied in various NLP tasks, including semantic parsing (Yih et al., 2014), search query retrieval (Shen et al., 2014), sentence modeling and classification (Kalchbrenner et al., 2014; Kim, 2014), name tagging and semantic role labeling (Collobert et al., 2011). For relation classification and extraction, there are two very recent works on CNNs for relation classification (Liu et al., 2013)[2] and (Zeng et al., 2014); however, to the best of our knowledge, there has been no work on employing CNNs for relation extraction so far. This paper is the first attempt to fill in that gap and serves as a baseline for future research in this area.

Our convolutional neural network is built upon that of Kalchbrenner et al. (2014) and Kim (2014) which are originally proposed for sentence classification and modeling. We adapt the network for relation extraction by introducing the position embeddings to encode the relative distances of the words in the sentence to the two entities of interest. Compared to the models in Liu et al. (2013) and Zeng et al. (2014) for relation classification that apply a single window size, our model for relation extraction incorporates various window sizes for convolutional filters, allowing the network to capture wider ranges of n-grams to be helpful for relation extraction. In addition, rather than initializing the word embeddings randomly as do Liu et al. (2013) and fixing the randomly generated position embeddings during training as do Zeng et al. (2014), we use pre-trained word embeddings for initialization and optimize both word embeddings and position embeddings as model parameters. More importantly, rather than using exterior features (either from human annotation or other pre-processing modules) to enrich the representation as do Liu et al. (2013) and Zeng et al. (2014), our model (adapted for RC where entity heads are given) avoids usage of manual linguistic resources and supervised NLP toolkits constructed externally, utilizing word embeddings that can be trained automatically in an unsupervised framework as the only external resource for the whole system.

We explore different model architectures systematically and demonstrate that the best model performance is achieved when multiple window sizes are implemented and the word embeddings, once initialized by some "universal" embeddings, are allowed to vary during the optimization process to reach an

[1]For evaluation purpose, we assume the positions of the two entities of interest in the sentence like most previous studies in this area (listed above). These are the only external features we need to achieve an end-to-end relation extractor.

[2]The title of the paper (Liu et al., 2013) on relation extraction is misleading since the authors actually do relation classification, according to the experimental description.

effective state for relation extraction. We evaluate our models on both relation classification and relation extraction tasks. For relation classification, experiments show that our model (without any external features and resources) outperforms the state-of-the-art models whether the external features are included in these models or not. For relation extraction, our model is significantly better than the baseline models that use the words and the embeddings themselves as the features. In the following, we discuss related work in Section 2 and present our model in Section 3. We detail an extensive evaluation in Section 4 and finally conclude in Section 5.

2 Related Work

As our present work focuses on the supervised framework for relation extraction, we concentrate on the supervised systems in this section. Besides the supervised systems (either feature-based or kernel-based) mentioned above, some recent systems have employed the *distant supervision* (DS) approach for relation extraction. This approach is essentially similar to the traditional systems in representing relation mentions but attempts to generate training data automatically by leveraging large knowledge bases of facts and corpus (Mintz et al., 2009; Riedel et al., 2010; Hoffmann et al., 2011; Surdeanu et al., 2012).

Regarding neural networks, their first application to NLP is language modeling which has been useful to learn distributed representations (embeddings) for words (Bengio et al., 2001; Mnih and Hinton, 2007; Collobert and Weston, 2008; Mnih and Hinton, 2009; Turian et al., 2010; Mikolov et al., 2013). These word embeddings have opened a new direction for many other NLP tasks grounded on neural networks. Some of them are mentioned above. Other than that, a class of recursive neural networks (RNNs) and neural tensor networks are proposed for paraphrase detection (Socher et al., 2011), parsing (Socher et al., 2013a), sentiment analysis (Socher et al., 2013b), knowledge base completion (Socher et al., 2013c), question answering (Mohit et al., 2014) etc. Among these RNN systems, the study that is most related to our relation extraction problem is Socher et al. (2012) that learns compositional vector representations for phrases and sentences through syntactic parse trees and applies these representations for relation classification. However, this method inherently requires syntactic parse trees in contrast to our target of avoiding use of any external features and resources for RC.

3 Convolutional Neural Network for Relation Extraction

Our convolutional neural network for relation extraction consists of four main layers: (i) the look-up tables to encode words in sentences by real-valued vectors, (ii) the convolutional layer to recognize n-grams, (iii) the pooling layer to determine the most relevant features and (iv) a logistic regression layer (a fully connected neural network with a softmax at the end) to perform classification (Collobert et al., 2011; Kim, 2014; Kalchbrenner et al., 2014). Figure 1 gives an overview of the network.

3.1 Word Representation

The input to the CNN for relation extraction consists of sentences marked with the two entity mentions of interest. As CNNs can only work with fixed length inputs, we compute the maximal separation between entity mentions linked by a relation and choose an input width greater than this distance. We insure that every input (relation mention) has this length by trimming longer sentences and padding shorter sentences with a special token.

Let n be the length of the relation mentions and $x = [x_1, x_2, \ldots, x_n]$ be some relation mention where x_i is the i-th word in the mention. Also, let x_{i_1} and x_{i_2} be the two heads of the two entity mentions of interest . Before entering the network, each word x_i is first transformed into a vector e_i by looking up the word embedding table $\mathbf{W}$ that can be initialized either by a random process or by some pre-trained word embeddings. Besides, in order to embed the positions of the two entity heads as well as the other words in the relation mention into the representation, for each word x_i, its relative distances to the two entity heads $i - i_1$ and $i - i_2$ are also mapped into real-value vectors d_{i_1} and d_{i_2} respectively using a position embedding table $\mathbf{D}$ (initialized randomly) (Collobert et al., 2011; Liu et al., 2013; Zeng et al., 2014). Note that the relative distances only range from $-n + 1$ to $n - 1$ so the position embedding matrix $\mathbf{D}$ has size $(2n - 1) \times m_d$ (m_d is a hyperparameter indicating the dimensionality of the position embedding vectors). Finally, the word embeddings

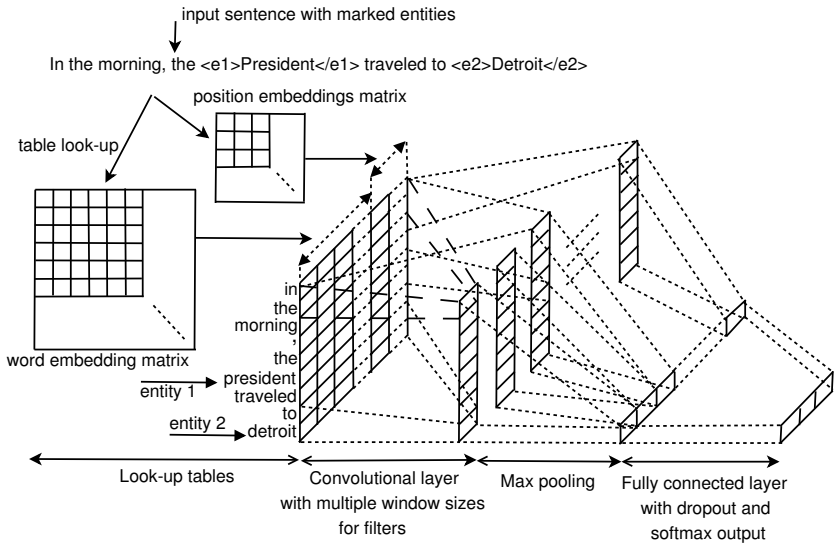

Figure 1: Convolutional Neural Network for Relation Extraction.

e_i and the position embeddings d_1 and d_2 are concatenated into a single vector $\mathbf{x}_i = [e_i, d_{i_1}, d_{i_2}]^\top$ to represent the word x_i. As a result, the original sentence x can now be viewed as a matrix $\mathbf{x}$ of size $(m_e + 2m_d) \times n$ where m_e is the dimensionality of the word embedding vectors.

$$\mathbf{x} = [\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n]$$

3.2 Convolution

In the next step, the matrix $\mathbf{x}$ representing the input relation mention is fed into the convolutional layer to extract higher level features. Given a widow size w, a filter is seen as a weight matrix $\mathbf{f} = [\mathbf{f}_1, \mathbf{f}_2, \ldots, \mathbf{f}_w]$ ($\mathbf{f}_i$ is a column vector of size $m_e + 2m_d$). The core of this layer is obtained from the application of the convolutional operator on the two matrices $\mathbf{x}$ and $\mathbf{f}$ to produce a score sequence $\mathbf{s} = [s_1, s_2, \ldots, s_{n-w+1}]$:

$$s_i = g(\sum_{j=0}^{w-1} \mathbf{f}_{j+1}^\top \mathbf{x}_{j+i}^\top + b)$$

where b is a bias term and g is some non-linear function. This process can then be replicated for various filters with different window sizes to increase the n-gram coverage of the model.

For relation extraction, we call the n-grams accompanied with relative positions of its words the augmented n-grams. It is instructive to think about the filter $\mathbf{f}$ as representing some hidden class of the augmented n-grams and the scores s_i as measuring the possibility the augmented n-gram at position i belongs to the corresponding hidden class (although these scores are not probabilities at all). The trained weights of the filter f would then amount to a feature detector that learns to recognize the hidden class of the augmented n-grams (Kalchbrenner et al., 2014).

3.3 Pooling

The rationale of the pooling layer is to further abstract the features generated from the convolutional layer by aggregating the scores for each filter to introduce the invariance to the absolute positions but preserve the relative positions of the n-grams between themselves and the entity heads at the same time. The popular aggregating function is $\max$ as it bears responsibility for identifying the most important or relevant features from the score sequence. Concretely, for each filter $\mathbf{f}$, its score sequence $\mathbf{s}$ is passed through the $\max$ function to produce a single number: $p_\mathbf{f} = \max\{\mathbf{s}\} = \max\{s_1, s_2, \ldots s_{n-w+1}\}$ which can be interpreted as estimating the possibility some augmented n-gram of the hidden class of $\mathbf{f}$ appears in the context.

3.4 Regularization and Classification

In the final step, the pooling scores for every filter are concatenated into a single feature vector $\mathbf{z} = [p_1, p_2, \ldots, p_m]$ to represent the relation mention. Here, m is the number of filters in the model and p_i is the pooling score of the i-th filter. Before actually applying this feature vector, following (Kim, 2014; Hinton et al., 2012), we execute a dropout for

regularization by randomly setting to zero a proportion ρ of the elements of the feature vector[3] $\mathbf{z}$ to produce the vector $\mathbf{z}_d$. The dropout vector $\mathbf{z}_d$ is then fed into a fully connected layer of standard neural networks followed by a softmax layer in the end to perform classification. The fully connected layer induces a weight matrix $\mathbf{C}$ as model parameters. At test time, the unseen relation mentions are scored using the feature vectors that are not dropped out. We also rescale the weights whose l_2-norms exceed a hyperparameter as Kim (2014).

Overall, the parameters for the presented CNN are: the word embedding matrix $\mathbf{W}$, the position embedding matrix $\mathbf{D}$, the m filter matrices, the weight matrix $\mathbf{C}$ for the fully connected layer. The gradients are computed using back-propagation while training is done via stochastic gradient descent with shuffled mini-batches and the AdaDelta update rule (Zeiler, 2012; Kim, 2014).

4 Experiments

4.1 Hyperparameters and Resources

For all the experiments below, we use: $\tanh$ for the non-linear function, 150 filters for each window size in the model and position embedding vectors with dimensionality of $m_d = 50$[4]. Regarding the other parameters, we use the same values as do Kim (2014), i.e, the dropout rate $\rho = 0.5$, the mini-batch size of 50, the hyperparameter for the l_2 of 3.

Finally, we utilize the pre-trained word embeddings `word2vec` from Mikolov et al. (2013) which have dimensionality of $m_e = 300$ and are trained on 100 billion words of Google News using the continuous bag-of-words architecture. These embeddings are publicly available here[5]. Vectors for the words not included in the pre-trained embeddings are initialized randomly. Besides the word embeddings `word2vec`, the model does not use any other NLP toolkits or resources.

4.2 Datasets

We evaluate our models on two datasets: the SemEval-2010 Task 8 dataset (Hendrickx et al., 2010) for relation classification and the ACE 2005 dataset for relation extraction.

ACE 2005 (87,512)		**SemEval 2010 (10,717)**	
Relation	%	Relation	%
ORG-AFF	2.8	Cause-Effect	12.4
PER-SOC	1.2	Component-Whole	11.7
ART	1.0	Entity-Destination	10.6
PART-WHOLE	1.4	Entity-Origin	9.1
GEN-AFF	1.1	Product-Producer	8.8
PHYS	2.1	Member-Collection	8.6
Other	**90.4**	Message-Topic	8.4
		Content-Container	6.8
		Instrument-Agency	6.2
		Other	**17.4**

Table 1: ACE 2005 and SemEval 2010 Relation Class Distributions

The SemEval dataset can be downloaded here[6] and contains 10,717 annotated examples, including 8,000 examples for training and 2,717 examples for testing. Each example is a sentence annotated for a pair of entities of interest and the corresponding relation class for this entity pair. There are 9 ordered relationships (with two directions) and an undirected *Other* class, resulting in 19 classes. A pair is counted as correct if the order of the entities in the relationship is correct. For the ACE 2005 dataset, documents are annotated for 6 major relation classes and 7 entity types. In order to generate the non-relation examples or the examples for the *Other* class, we collect every pair of entity mentions within a single sentence and not included in the annotated relation set. To reduce the noise, we truncate the generated dataset by removing all the examples whose distances between the two entity heads are greater than 15. This results in a considerably unbalanced dataset of 8,365 positive examples of the 6 annotated relation classes and 79,147 negative examples of the class *Other*. The distributions of the relation classes on the two datasets are shown in Table 1. As we can see, the ACE dataset is much more biased toward the *Other* class than the SemEval dataset and thus more appropriate for relation extraction experiments.

4.3 Evaluation of Model Architectures

We investigate the effectiveness of different window sizes of filters by running the proposed CNN

[3]Following the Bernoulli distribution

[4]These values produce the best performance during our experimental process.

[5]https://code.google.com/p/word2vec/

[6]http://docs.google.com/View?id=dfvxd49s 36c28v9pmw

		nonstatic.rand			static.word2vec			nonstatic.word2vec		
#	window sizes	P	R	F	P	R	F	P	R	F
1	2	69.56	41.64	52.04	74.66	41.03	52.90	72.74	49.49	58.87
2	3	68.47	42.73	52.57	74.19	42.16	53.73	72.50	50.75	59.66
3	4	68.17	43.39	52.94	73.60	41.90	53.35	72.56	49.81	58.97
4	5	66.83	43.46	52.55	73.52	42.60	53.89	71.70	51.08	59.57
5	4-5	66.18	46.12	54.25	72.69	45.23	55.71	71.88	52.36	60.50
6	3-4-5	67.54	45.73	54.43	71.99	46.85	56.73	71.21	53.24	60.86
7	2-3-4-5	66.42	47.20	55.12	72.60	46.77	56.85	71.25	53.91	**61.32**

Table 2: System Performance on various window size combinations and architectures

model on window sizes of 2, 3, 4 and 5. To understand the behavior of the model on multiple window sizes, we further test it on the following window size combinations: (4,5), (3,4,5) and (2,3,4,5). In each of these window size configurations, we evaluate the system on three different scenarios: (i) the word embeddings and the position embeddings are randomly initialized and optimized during the training process (denoted by *nonstatic.rand*), (ii) the word embeddings are initialized by the pre-trained word embeddings; the position embeddings are initialized randomly and the two embeddings are kept unchanged during the training (denoted by *static.word2vec*), (iii) the two embeddings are initialized as in case (ii) but they are optimized as model parameters when the model is trained (denoted by *nonstatic.word2vec*). These experiments are carried out for relation extraction on the ACE 2005 dataset via 5-fold cross validation. Table 2 presents the system performance on Precision (P), Recall (R) and F1 score (F).

The key observations from the table are[7]:

(i) From rows 1, 2, 3, 4, we see that evaluating window sizes individually is quite intricate. It is unclear which window size is the best size for CNNs on relation extraction. For instance, on the *nonstatic.rand* mode, the window size 4 seems to outperform the others while on the other modes, the window sizes 3 and 5 turn out to be better. Besides, the performance gaps between the window sizes are small, making it hard to draw a conclusive judgement. In any case, the window size 2 seems to be the worst, suggesting that the 2-grams might be less informative than the others on representing relation mentions for CNNs on this dataset.

(ii) While the results on evaluating single window sizes are hard to analyze, the results for multiple window sizes are quite clear and conclusive. Moving from single window sizes of 2, 3, 4 or 5 (rows 1, 2, 3 and 4 respectively) to the configuration with two window sizes 4 and 5 (row 5) gives us consistent improvements on all the model architectures. The performance is then consistently enhanced when more window sizes are included, resulting in the best performance when all the window sizes 2, 3, 4 and 5 are employed. This demonstrates the advantages of the models with multiple window sizes over the single window size models in Liu et al. (2013) and Zeng et al. (2014).

(iii) Regarding different model architectures, the picture is even clearer. No matter which window size configuration is applied, we constantly see the *nonstatic.word2vec* architecture performs most effectively, followed by the *static.word2vect* setting which is in turn followed by the *nonstatic.rand* model. This suggests the undeniable benefits of initializing the word embeddings by some "universal" pre-trained values and updating the embeddings to reflex RE specific embeddings when training the models (Collobert et al., 2011; Kim, 2014). For the next experiments, we always use all the window sizes 2, 3, 4 and 5 with the *nonstatic.word2vec* architecture.

4.4 Relation Extraction Experiment

We compare our system with the traditional feature-based relation extraction systems when these system are only allowed to use the same information and resources as our systems, i.e, the words in the relation mentions, the positions of the two entity heads and the word embeddings. Given the sentences and the positions of the two entity heads, the features that the state-of-the-art feature-based systems extract in-

[7]The statements at points (ii) and (iii) are significant at confidence levels $\geq 95\%$.

clude: the heads of the two entity mentions; the words in the context before mention 1; after mention 2 and between two mentions; the bigrams, the word sequences between two entities, the order of two mentions, the number of words between two mentions (Zhou et al., 2005; Jiang and Zhai, 2007; Sun et al., 2011). The feature-based system using this feature set is called *Words*. Armed with the word embeddings, one can further introduce these embeddings into the head words or the words in the context as additional features (Nguyen and Grishman, 2014). We call the system *Words* augmented with the embeddings for the two heads *Words-HM-Wed* and *Words* augmented with the embeddings for words in the contexts *Words-WC-Wed*. We apply the MaxEnt framework with L2 regularization in the Mallet toolkit[8] to train these feature-based models (as (Jiang and Zhai, 2007; Sun et al., 2011; Nguyen and Grishman, 2014)). Table 3 shows the performance of the three baseline systems and our proposed CNN via 5-fold cross validation on the ACE 2005 dataset.

System	P	R	F
Words	54.95	43.73	48.69
Words-WC-Wed	50.10	44.47	47.11
Words-HM-Wed	57.01	55.74	56.36
Our CNN	71.25	53.91	**61.32**

Table 3: Performance of Relation Extraction Systems

The first observation is that adding the word embeddings to the words in the context hurt the performance of the feature-based systems while augmenting the heads of the entities with word embeddings significantly improves the feature-based systems. This is consistent with the results reported by Nguyen and Grishman (2014) and demonstrates that the ability to wisely pick the words for embeddings and avoid embeddings on specific locations is crucial to the feature-based systems. More importantly, our proposed CNN significantly outperforms all the baseline models at the confidence levels $\geq 95\%$, an improvement of 4.96% over the best feature-based system *Words-HM-Wed* (Nguyen and Grishman, 2014). This result indicates that CNNs are a better way to employ word embeddings for relation extraction.

[8]http://mallet.cs.umass.edu/

Remember that although the traditional systems can achieve a performance greater than 72% on the ACE dataset (Qian et al., 2008; Sun et al., 2011), they come at the expense of elaborate feature engineering as well as much more expensive feature extraction. In particular, the feature extractors of these feature-based systems require: (i) the perfect entity and mention type information hand-labeled laboriously by human annotators; (ii) the extensive usage of the existing supervised NLP toolkits and resources (constituent and dependency parsers, dictionaries, gazetteers etc) which might be unavailable for various domains in reality. The absence of the perfect (hand-annotated) entity and mention type information (i.e point (i) above) greatly impairs these feature-based systems' performance. For instance, both Plank and Moschitti (2013) and Nguyen and Grishman (2014) report a performance less than 60% on the ACE 2005 dataset when the perfect entity type and mention type features are not employed although the other features with extensive feature engineering (i.e point (ii) above) are still included. As a result, in a more realistic setting where hand-annotated features are prohibitive, the proposed CNN requires much less feature engineering and resources but still performs better than the traditional feature-based systems.

4.5 Relation Classification Experiment

In order to further verify the effectiveness of the system, we test the system on the relation classification task with the SemEval 2010 dataset and compare the results with the state-of-the-art systems in this area. Table 4 describes the performance of various traditional systems that are based on classifiers such as MaxEnt and SVM with series of supervised and manual features[9](Hendrickx et al., 2010) as well as the more recent systems based on convolutional neural networks (Zeng et al., 2014) (O-CNN), recursive neural networks (RNN), matrix-vector recursive neural networks (MVRNN) (Socher et al., 2012) or log-quadratic factor-based compositional embedding model (FCM) (Yu et al., 2014)[10].

As we can see, among the systems *not using any*

[9]i.e the features extracted from supervised pre-processing NLP modules and manual resources

[10]These are the macro-averaged F1-scores, computed by the officially provided scorer.

Classifier	Feature Sets	F
SVM	POS, WordNet, morphological features, thesauri, Google n-grams	77.6
MaxEnt	POS, WordNet, morphological features, noun compound system, thesauri, Google n-grams	77.6
SVM	POS, WordNet, prefixes and other morphological features, dependency parse, Levin classes, PropBank, FrameNet, NomLex-Plus, Google n-grams, paraphrases, TextRunner	82.2
RNN	-	74.8
RNN	POS, name tagging, WordNet	77.6
MVRNN	-	79.1
MVRNN	POS, name tagging, WordNet	82.4
O-CNN	-	78.9
O-CNN	WordNet	82.7
FCM	-	80.6
FCM	dependency parse, name tagging	83.0
Our CNN	-	**82.8**

Table 4: Performance of Relation Classification Systems

supervised and manual features (i.e, POS, WordNet, name tagging, dependency parse, patterns etc), our system significantly outperforms the state-of-the-art system FCM (80.6%) (Yu et al., 2014) with an improvement of 2.2%. More interestingly, even without supervised and manual features, our system can still work comparably to the other systems utilizing these features as the vital components. For instance, the supervised features (dependency parse and name tagging) are crucial to FCM (Yu et al., 2014) to significantly improve its performance. We attribute our performance advantage over the closely-related system O-CNN (Zeng et al., 2014) to the multiple window sizes, the optimization of the position embeddings during training and possibly the superiority of the embeddings *word2vec* we use.

4.6 Impact of Unbalanced Dataset

Shifting from relation classification to relation extraction with an unbalanced corpus, we witness a large performance gap as described above. In this section, we study the impact of the unbalanced corpus on the performance of relation extractors for both convolutional neural networks and traditional feature-based approaches (*Words* and *Words-HM-Wed*). In particular, we vary the ratio of positive (true relations) and negative (the class *Other*) examples in the ACE 2005 dataset and see how the system performance responds to this variation. Figure 2 shows the curves. This is a 5-fold cross validation experiment and all the comparisons are significant at confidence levels $\geq 95\%$.

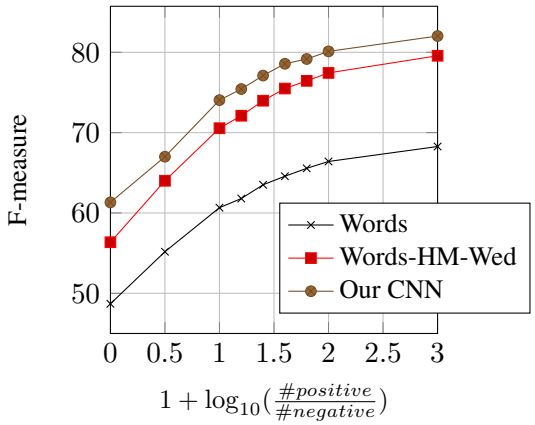

Figure 2: F measures vs positive/negative ratios

From the figure, we see that all the models improve constantly with the increase of the ratio of the positive and negative examples. The performance peaks with an improvement of about 20% for all models when the number of examples of the class *Other* is small relative to the others. In other words, the systems attain their best performance when relation extraction is reduced to the relation classification problem, suggesting that relation extraction is much more challenging than relation classification. Finally, for all the ratio values, we consistently see that the convolutional neural network is superior to the others, once again confirming its advantages.

5 Conclusion

We present a CNN for relation extraction that emphasizes an unbalanced corpus and minimizes usage of external supervised NLP toolkits for features. The network uses multiple window sizes for filters, position embeddings for encoding relative distances and pre-trained word embeddings for initialization in a non-static architecture. The experimental results demonstrate the effectiveness of the proposed CNN on both RC and RE. Our future work includes: (i) to enrich the representation of CNNs with more features for RE, (ii) to study the applications of CNNs on other related tasks, and (iii) to examine other neural network models for RE.

References

Yoshua Bengio, Réjean Ducharme, and Pascal Vincent. 2001. *A Neural Probabilistic Language Model*. In Advances in Neural Information Processing Systems 13 (NIPS'00), pages 932938, MIT Press, 2001.

John Blitzer, Ryan McDonald, and Fernando Pereira. 2006. *Domain Adaptation with Structural Correspondence Learning*. In Proceedings of the 2006 Conference on Empirical Methods in Natural Language Processing, Sydney, Australia.

Elizabeth Boschee, Ralph Weischedel, and Alex Zamanian. 2005. *Automatic Information Extraction*. In Proceedings of the International Conference on Intelligence Analysis.

Razvan C. Bunescu and Raymond J. Mooney. 2005a. *A Shortest Path Dependency Kenrel for Relation Extraction*. In Proceedings of HLT/EMNLP.

Razvan C. Bunescu and Raymond J. Mooney. 2005b. *Subsequence Kernels for Relation Extraction*. In Proceedings of NIPS.

Yee S. Chan and Dan Roth. 2010. *Exploiting Background Knowledge for Relation Extraction*. In Proceedings of the 23rd International Conference on Computational Linguistics (Coling 2010), pages 152-160, Beijing, China, August.

Ronan Collobert and Jason Weston. 2008. *A Unied Architecture for Natural Language Processing: Deep Neural Networks with Multitask Learning*. In International Conference on Machine Learning, ICML, 2008.

Ronan Collobert, Jason Weston, Léon Bottou, Michael Karlen, Koray Kavukcuoglu and Pavel Kuksa. 2011. *Natural Language Processing (Almost) from Scratch*. Journal of Machine Learning Research 12:24932537.

Aron Culotta and Jeffrey Sorensen. 2004. *Dependency Tree Kernels for Relation Extraction*. In Proceedings of ACL 2004.

Hal Daumé III. 2007. *Frustratingly Easy Domain Adaptation*. In Proceedings of the ACL, pages 256-263, Prague, Czech Republic, June 2007.

Ralph Grishman, David Westbrook and Adam Meyers. 2005. *NYUs English ACE 2005 System Description*. ACE 2005 Evaluation Workshop.

Jing Jiang and ChengXiang Zhai. 2007. *A Systematic Exploration of the Feature Space for Relation Extraction*. In Proceedings of the Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics (NAACL-HLT'07), pages 113-120, 2007.

Iris Hendrickx, Su Nam Kim, Zornitsa Kozareva, Preslav Nakov, Diarmuid Ó Séaghdha, Sebastian Padó, Marco Pennacchiotti, Lorenza Romano, Stan Szpakowicz 2010. *SemEval-2010 Task 8: Multi-Way Classification of Semantic Relations Between Pairs of Nominals*. In Proceedings of the 5th International Workshop on Semantic Evaluation, SemEval 2010.

Geoffrey E. Hinton, Nitish Srivastava, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2012. *Improving Neural Networks by Preventing Co-Adaptation of Feature Detectors*. CoRR, abs/1207.0580.

Raphael Hoffmann, Congle Zhang, Xiao Ling, Luke Zettlemoyer, and Daniel Weld. 2011. *Knowledge-based Weak Supervision for Information Extraction of Overlapping Relations*. In Proceedings of ACL 2011.

Nal Kalchbrenner, Edward Grefenstette and Phil Blunsom. 2014. *A Convolutional Neural Network for Modelling Sentences*. In Proceedings of ACL 2014.

Nanda Kambhatla. 2004. *Combining Lexical, Syntactic, and Semantic Features with Maximum Entropy Models for Information Extraction*. In Proceedings of ACL 2004.

Yoon Kim. 2014. *Convolutional Neural Networks for Sentence Classification*. In Proceedings of EMNLP 2014.

Yann LeCun, Léon Bottou, Yoshua Bengio and Patrick Haffner. 1988. *Gradient-based learning applied to document recognition*. In Proceedings of the IEEE, 86(11):22782324, November, 1988.

ChunYang Liu, WenBo Sun, WenHan Chao, and WanXiang Che. 2013. *Convolution Neural Network for Relation Extraction*. In Proceedings of 9th International Conference on Advanced Data Mining and Applications, Part II (ADMA 2013), Hangzhou, China, December, 2013.

David McClosky, Eugene Charniak, and Mark Johnson. 2010. *Automatic Domain Adaptation for Parsing*. In Proceedings of Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics, pages 28-36, Los Angeles, California, June 2010.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. *Distributed Representations of Words and Phrases and their Compositionality*. In Proceedings of NIPS, 2013.

Mike Mintz, Steven Bills, Rion Snow, and Daniel Jurafsky. 2009. *Distant Supervision for Relation Extraction without Labeled Data*. In Proceedings of ACL, 2009.

Andriy Mnih and Geoffrey Hinton. 2007. *Three new Graphical Models for Statistical Language Modelling*. In Proceedings of ICML'07, pages 641-648, Corvallis, OR, 2007.

Andriy Mnih and Geoffrey Hinton. 2009. *A Scalable Hierarchical Distributed Language Model*. In NIPS, page 1081-1088.

Mohit Iyyer, Jordan Boyd-Graber, Leonardo Claudino, Richard Socher and Hal Daumé III. 2014. *A Neural*

Network for Factoid Question Answering over Paragraphs. In Proceedings of EMNLP 2014.

Thien Huu Nguyen and Ralph Grishman. 2014. *Employing Word Representations and Regularization for Domain Adaptation of Relation Extraction*. In Proceedings of ACL 2014, pages 68-74, Baltimore, Maryland, USA.

Truc-Vien T. Nguyen, Alessandro Moschitti, and Giuseppe Riccardi. 2009. *Convolution Kernels on Constituent, Dependency and Sequential Structures for Relation Extraction*. In Proceedings of EMNLP 09, pages 1378-1387, Stroudsburg, PA, USA.

Barbara Plank and Alessandro Moschitti. 2013. *Embedding Semantic Similarity in Tree Kernels for Domain Adaptation of Relation Extraction*. In Proceedings of the ACL 2013, pages 1498-1507, Sofia, Bulgaria.

Longhua Qian, Guodong Zhou, Qiaoming Zhu and Peide Qian. 2008. *Exploiting Constituent Dependencies for Tree Kernel-based Semantic Relation Extraction*. In Proceedings of COLING, pages 697-704, Manchester.

Sebastian Riedel, Limin Yao, and Andrew McCallum. 2010. *Modeling Relations and their Mentions without Labeled Text*. In Proceedings of ECML PKDD, 2010.

Yelong Shen, Xiaodong He, Jianfeng Gao, Li Deng and Grégoire Mesnil. 2014. *Learning Semantic Representations Using Convolutional Neural Networks for Web Search*. In Proceedings of WWW 2014.

Richard Socher, Eric H. Huang, Jeffrey Pennington, Andrew Y. Ng, and Christopher D. Manning. 2011. *Dynamic Pooling and Unfolding Recursive Autoencoders for Paraphrase Detection*. In Proceedings of NIPS 2011.

Richard Socher, Brody Huval, Christopher D. Manning and Andrew Y. Ng. 2012. *Semantic Compositionality through Recursive Matrix-Vector Spaces*. In Proceedings of EMNLP 2012.

Richard Socher, John Bauer, Christopher D. Manning and Andrew Y. Ng. 2013. *Parsing with Compositional Vector Grammars*. In Proceedings of ACL 2013.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Chris Manning, Andrew Ng and Chris Potts. 2013. *Recursive Deep Models for Semantic Compositionality Over a Sentiment Treebank*. In Proceedings of EMNLP 2013.

Richard Socher, Danqi Chen, Christopher D. Manning, Andrew Y. Ng. 2013. *Reasoning With Neural Tensor Networks for Knowledge Base Completion*. In Proceedings of NIPS 2013.

Ang Sun, Ralph Grishman, and Satoshi Sekine. 2011. *Semi-supervised Relation Extraction with Large-scale Word Clustering*. In Proceedings of ACL-HLT, pages 521-529, Portland, Oregon, USA.

Le Sun and Xianpei Han 2014. *A Feature-Enriched Tree Kernel for Relation Extraction*. In Proceedings of ACL 2014, pages 61-67, Baltimore, Maryland, USA.

Mihai Surdeanu, Julie Tibshirani, Ramesh Nallapati and Christopher D. Manning 2012. *Multi-instance Multi-label Learning for Relation Extraction*. In Proceedings of EMNLP-CoNLL 2012.

Joseph Turian, Lev Ratinov, and Yoshua Bengio. 2010. *Word representations: A simple and general method for semi-supervised learning*. In Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics (ACL'10), pages 384-394, Uppsala, Sweden, July, 2010.

Wen-tau Yih, Xiaodong He, and Christopher Meek. 2014. *Semantic Parsing for Single-Relation Question Answering*. In Proceddings of ACL 2014.

Mo Yu, Matthew R. Gormley, and Mark Dredze. 2014. *Factor-based Compositional Embedding Models*. In the NIPS Learning Semantics Workshop 2014.

Matthew D. Zeiler. 2012. *ADADELTA: An Adaptive Learning Rate Method*. CoRR, abs/1212.5701.

Dmitry Zelenko, Chinatsu Aone, and Anthony Richardella. 2003. *Kernel Methods for Relation Extraction*. Journal of Machine Learning Research, 3:10831106.

Daojian Zeng, Kang Liu, Siwei Lai, Guangyou Zhou and Jun Zhao. 2014. *Relation Classification via Convolutional Deep Neural Network*. In Proceedings of COLING 2014.

Min Zhang, Jie Zhang, Jian Su, and GuoDong Zhou. 2006. *A Composite Kernel to Extract Relations between Entities with both Flat and Structured Features*. In Proceedings of COLING-ACL-06, pages 825-832, Sydney.

Guodong Zhou, Jian Su, Jie Zhang, and Min Zhang. 2005. *Exploring various Knowledge in Relation Extraction*. In Proceedings of ACL'05, pages 427-434, Ann Arbor, USA, 2005.

Guodong Zhou, Min Zhang, DongHong Ji, and QiaoMing Zhu. 2007. *Tree Kernel-based Relation Extraction with Context-sensitive Structured Parse Tree Information*. In Proceedings of EMNLP-CoNLL-07, pages 728-736, Prague.