LABDA at SemEval-2017 Task 10: Relation Classification between keyphrases via Convolutional Neural Network

Víctor Suárez-Paniagua, Isabel Segura-Bedmar and Paloma Martínez
Computer Science Department
Carlos III University of Madrid
Leganés 28911, Madrid, Spain
vspaniag,isegura,pmf@inf.uc3m.es

Abstract

In this paper, we describe our participation at the subtask of extraction of relationships between two identified keyphrases. This task can be very helpful in improving search engines for scientific articles. Our approach is based on the use of a convolutional neural network (CNN) trained on the training dataset. This deep learning model has already achieved successful results for the extraction relationships between named entities. Thus, our hypothesis is that this model can be also applied to extract relations between keyphrases. The official results of the task show that our architecture obtained an F1-score of 0.38% for Keyphrases Relation Classification. This performance is lower than the expected due to the generic preprocessing phase and the basic configuration of the CNN model, more complex architectures are proposed as future work to increase the classification rate.

1 Introduction

Nowadays, a deluge of scientific articles is published every year, which demonstrates that we are living in an emerging knowledge era. An important drawback of this situation is that the study of a given field or problem requires reviewing an huge number of scientific publications, becoming such a very arduous task. Most search engines apply linguistic normalization (such as lemmatization or stemming) and some of them also exploit the semantic analysis of texts in order to detect concepts to improve their recall. The goal of the ScienceIE Task at SemEval 2017 (Augenstein et al., 2017) is the extraction of keyphrases (such as MATERIALS, PROCESSES and TASKS) and relationships between them from scientific articles. This competition provides a common evaluation framework to researches allowing a fair way to evaluate and compare their approaches. Our participation focuses on the subtask of extracting relationships between keyphrases. In particular, these relationships are HYPONYM-OF (for example, 'femur' is HYPONYM-OF 'bone'), SYNONYM-OF (for example, 'ophthalmologist' is SYNONYM-OF 'oculist'), and NONE. The detection of these relationships between keyphrases can improve the performance of current researches.

In this paper, we describe the participation of the group LaBDA for participating in the subtask C (extraction of relationships between keyphrases) evaluated on the scenario 3, where the test dataset includes texts as well as the annotations of their keyphrases (boundaries and types). Our approach is based on the CNN proposed in (Kim, 2014), which was the first work to exploit a CNN for the task of sentence classification. This model was able to infer the class of each sentence, and returned good results without the need for external information. To this end, the model computes an output vector, which describes the whole sentence, and applies convolving filters to the input through several windows of different sizes. Finally, this vector is used in a classification layer to assign a class label. Recently, CNN has succeeded providing the state-of-art results in some tasks of relation extraction such as the relationships between nominals (Zeng et al., 2014) or the extraction of drug-drug interactions (Zhao et al., 2016). Our aim is to explore if CNN is also a suitable method for extracting relationships between keyphrases.

2 Dataset

The valuable contribution of the ScienceIE challenge was to provide an annotated corpus for train-

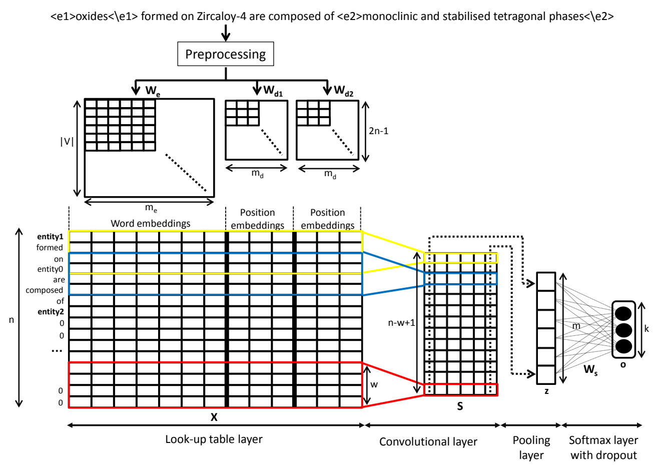

Figure 1: CNN model for the ScienceIE keyphrases Relation Classification task of SemEval 2017.

ing and evaluating supervised algorithms to extract Keyphrases from Scientific Publications. The whole corpus contains 500 journal articles about Computer Science, Material Sciences and Physics from ScienceDirect[1]. The corpus is split into training, development and testing sets with 350, 50 and 100 documents, respectively. A detailed description of the method used to collect and process documents can be found in (Augenstein et al., 2017).

2.1 Pre-processing phase

Each pair of keyphrases represents a possible relation instance. Each of these instances is classified by the CNN model in three classes HYPONYM-OF, SYNONYM-OF and NONE. The corpus is given in the paragraph level, that is why we use the NLTK[2] sentence splitter to separate the relations in the sentence level because we only have to annotate relations within a sentence.

Once we have each instance, following a similar approach as that described in (Kim, 2014), the sentences were tokenized and cleaned (converting all words to lower-case and separating special characters with white spaces by regular expressions). Then, the two keyphrases of each instance were replaced by the labels ”entity1” and ”entity2”, and by ”entity0” for the remaining keyphrases in the sentence. This method is known as entity blinding, and verifies the generalization of the model.

In the case of the HYPONYM-OF class the directionality must be considered. For instance, if an entity A is HYPONYM-OF B we annotated the relation of B with A as NONE. For the remainder classes, we annotated both directions with the same class label.

The keyphrases corpus contains a number of overlapped keyphrases. As this kind of mentions produces bad entity blinding, we decided to remove them. The classification of keyphrases involving overlapped entities is a challenging task which will be tackled in future work. One possible solution will consider different instances for each overlapped entities.

3 CNN model

In this section, we present a CNN model for the special case of sentences which describe keyphrases relationships. Figure 1 shows the whole process from its input, which is a sentence with marked entities, until the output, which is the classification of the instance into one of the keyphrases relation types.

3.1 Word table layer

After the pre-processing phase, we created an input matrix suitable for the CNN architecture. The input matrix should represent all training instances for the CNN model; therefore, they should have the same length. We determined the maximum length of the sentence in all the instances (denoted by n), and then extended those sentences with lengths shorter than n by padding with an auxiliary token ”*0*”.

Moreover, each word has to be represented by a vector. To do this, we considered to randomly initialize a vector for each different word which allows us to replace each word by its word embedding vector: $\mathbf{W}_e \in \mathbb{R}^{|V| \times m_e}$ where V is the

[1]http://www.sciencedirect.com/
[2]http://www.nltk.org

vocabulary size and m_e is the word embedding dimension. Finally, we obtained a vector $\mathbf{x} = [x_1; x_2; ...; x_n]$ for each instance where each word of the sentence is represented by its corresponding word vector from the word embedding matrix. We denote p_1 and p_2 as the positions in the sentence of the two entities to be classified.

The following step involves calculating the relative position of each word to the two interacting keyphrases as $i - p_1$ and $i - p_2$, where i is the word position in the sentence (padded word included), in the same way as (Zeng et al., 2014). In order to avoid negative values, we transformed the range $(-n+1, n-1)$ to the range $(1, 2n-1)$. Then, we mapped these distances into a real value vector using two position embedding $\mathbf{W}_{d1} \in \mathbb{R}^{(2n-1)\times m_d}$ and $\mathbf{W}_{d2} \in \mathbb{R}^{(2n-1)\times m_d}$. Finally, we created an input matrix $\mathbf{X} \in \mathbb{R}^{n\times(m_e+2m_d)}$ which is represented by the concatenation of the word embeddings and the two position embeddings for each word in the instance.

3.2 Convolutional layer

Once we obtained the input matrix, we applied a filter matrix $\mathbf{f} = [f_1; f_2; ...; f_w] \in \mathbb{R}^{w\times(m_e+2m_d)}$ to a context window of size w in the convolutional layer to create higher level features. For each filter, we obtained a score sequence $\mathbf{s} = [s_1; s_2; ...; s_{n-w+1}] \in \mathbb{R}^{(n-w+1)\times 1}$ for the whole sentence as

$$s_i = g(\sum_{j=1}^{w} f_j x_{i+j-1}^T + b)$$

where b is a bias term and g is a non-linear function (such as tangent or sigmoid). Note that in Figure 1, we represent the total number of filters, denoted by m, with the same size w in a matrix $\mathbf{S} \in \mathbb{R}^{(n-w+1)\times m}$. However, the same process can be applied to filters with different sizes by creating additional matrices that would be concatenated in the following layer.

3.3 Pooling layer

Here, the goal is to extract the most relevant features of each filter using an aggregating function. We used the max function, which produces a single value in each filter as $z_f = max\{\mathbf{s}\} = max\{s_1; s_2; ...; s_{n-w+1}\}$. Thus, we created a vector $\mathbf{z} = [z_1, z_2, ..., z_m]$, whose dimension is the total number of filters m representing the relation instance. If there are filters with different sizes, their output values should be concatenated in this layer.

3.4 Softmax layer

Prior to performing the classification, we performed a dropout to prevent overfitting. We obtained a reduced vector $\mathbf{z}_d$, randomly setting the elements of $\mathbf{z}$ to zero with a probability p following a Bernoulli distribution. After that, we fed this vector into a fully connected softmax layer with weights $\mathbf{W}_s \in \mathbb{R}^{m\times k}$ to compute the output prediction values for the classification as $\mathbf{o} = \mathbf{z}_d\mathbf{W}_s + d$ where d is a bias term; we have $k = 3$ classes in the dataset (HYPONYM-OF, SYNONYM-OF and NONE). At test time, the vector $\mathbf{z}$ of a new instance is directly classified by the softmax layer without a dropout.

3.5 Learning

For the training phase, we need to learn the CNN parameter set θ = ($\mathbf{W}_e$, $\mathbf{W}_{d1}$, $\mathbf{W}_{d2}$, $\mathbf{W}_s$, d, $\mathbf{F}_m$, b), where $\mathbf{F}_m$ are all of the m filters $\mathbf{f}$. For this purpose, we used the conditional probability of a relation r obtained by the softmax operation as

$$p(r|\mathbf{x}, \theta) = \frac{\exp(\mathbf{o}_r)}{\sum_{l=1}^{k} \exp(\mathbf{o}_l)}$$

to minimize the cross entropy function for all instances ($\mathbf{x}_i$,y_i) in the training set T as follows

$$J(\theta) = \sum_{i=1}^{T} \log p(y_i|\mathbf{x}_i, \theta)$$

In addition, we minimize the objective function by using stochastic gradient descent over shuffled mini-batches and the Adam update rule (Kingma and Ba, 2014) to learn the parameters.

4 Results and Discussion

Firstly, we use a basic CNN predefined parameters to create a baseline system without a position embeddings. Secondly, the effects of the position embeddings were observed assigning a dimension $M_d = 10$. In addition, we define the parameters the same as in (Kim, 2014): word embeddings dimension $M_e = 300$, filters $m = 200$ with a window size $w = (3, 4, 5)$. For the training phase we use: a dropout rate $p = 50\%$, mini-batch size of size 50 and the Rectified Linear Unit (ReLU) as the non-linear function g. The parameter $n = 95$ which is determined by the maximum length sentence in the dataset. Only the number of epochs was fine-tuned in the validation set using the stopping criteria.

The results of the basic CNN configuration without position embeddings are showed in Table 1. We observe that the Recall performance in both classes are very low, the reason is that the parameters were not explored in detail and the model does not fit to this problem. In addition, the entities overlapped removal discard many examples that can improve the results.

	Precision	Recall	F1-score
HYPONYM-OF	0.27	0.07	0.12
SYNONYM-OF	0.65	0.32	0.43
Total	0.53	0.21	0.30

Table 1: Results over the dataset using a basic CNN.

The official results obtained by the CNN with position embeddings are showed in Table 2. We observe that the Precision increases considerably in the case of HYPONYM-OF and the Recall of SYNONYM-OF. This proves that sentences are best represented using position embeddings (+8% in F1-score against to CNN without position embeddings).

For both cases, the class SYNONYM-OF is classified better than the class HYPONYM-OF because the examples in the former class are very clear, e.g. in the sentence ”trajectory surface hoping (TSH)” the keyphrase ”trajectory surface hoping” is a SYNONYM-OF ”TSH”, and, also, we obtained the double of instances due to the class is the same in both directions. That is the main reason why the class HYPONYM-OF obtained low Recall in both models. For this reason, we will add some preprocessing to correct these classification errors with a rule-based system.

	Precision	Recall	F1-score
HYPONYM-OF	0.54	0.07	0.13
SYNONYM-OF	0.61	0.46	0.52
Total	0.60	0.28	0.38

Table 2: Results over the dataset using a CNN with position embedding size of 10.

5 Conclusions and Future work

We present the CNN model used by LaBDA Team for the ScienceIE keyphrases Relation Classification task of SemEval 2017. We find that the performance of the CNN model in this task is promising but the results are lower in comparison with other participant results. However, we only try a basic configuration with a generic preprocessing phase and without external features. The results suggest that the usage of the position embedding improves the performance in both classes.

As future work we will explore and fine-tune all the parameters of this architecture such as the size of the position embeddings, the number and the size of the filters. In addition, we will tackle the overlapped entities problem using each entity as different instances. Thus, more examples will be used for each class to increase the classification rate. Furthermore, we will train a CNN with different pre-trained word embedding models instead of using a random word embedding initialization and comparing results with and without position embeddings.

Funding

This work was supported by the Research Program of the Ministry of Economy and Competitiveness - Government of Spain, (eGovernAbility-Access project TIN2014-52665-C2-2-R).

References

Isabelle Augenstein, Mrinal Das, Sebastian Riedel, Lakshmi Vikraman, and Andrew McCallum. 2017. Semeval 2017 task 10: Scienceie - extracting keyphrases and relations from scientific publications. In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*. Association for Computational Linguistics, Vancouver, Canada, pages 546–555. http://www.aclweb.org/anthology/S17-2091.

Y. Kim. 2014. Convolutional neural networks for sentence classification. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. pages 1746–1751.

Diederik P. Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *CoRR* abs/1412.6980.

Daojian Zeng, Kang Liu, Siwei Lai, Guangyou Zhou, and Jun Zhao. 2014. Relation classification via convolutional deep neural network. In *Proceedings of the 25th International Conference on Computational Linguistics (COLING 2014), Technical Papers*. Dublin City University and Association for Computational Linguistics, Dublin, Ireland, pages 2335–2344.

Zhehuan Zhao, Zhihao Yang, Ling Luo, Hongfei Lin, and Jian Wang. 2016. Drug drug interaction extraction from biomedical literature using syntax convolutional neural network. *Bioinformatics* .