UniMa at SemEval-2018 Task 7: Semantic Relation Extraction and Classification from Scientific Publications

Thorsten Keiper,[1] Zhonghao Lyu,[1] Sara Pooladzadeh,[1] Yuan Xu,[1] Jingyi Zhang,[1]
Anne Lauscher,[1,2] and **Simone Paolo Ponzetto[1]**
[1]University of Mannheim, Germany
[2]Stuttgart Media University, Germany
{tkeiper, zlyu, spooladz, yuaxu, jizhang}@mail.uni-mannheim.de
{anne, simone}@informatik.uni-mannheim.de

Abstract

Large repositories of scientific literature call for the development of robust methods to extract information from scholarly papers. This problem is addressed by the SemEval 2018 Task 7 on extracting and classifying relations found within scientific publications. In this paper, we present a feature-based and a deep learning-based approach to the task and discuss the results of the system runs that we submitted for evaluation.

1 Introduction

Nowadays, the exploding amount of scientific literature makes it ever more problematic for researchers and scholars to get focused access to the state-of-the-art in a certain field of science. Therefore, it is getting increasingly important to develop effective computational approaches for extracting information from large scholarly corpora. SemEval 2018 Task 7 (Gábor et al., 2018) addresses this problem with a shared task on extracting and classifying semantic relations in scientific papers. The task is divided into two subtasks:

1. **Relation Classification.** Given an existing relation between two entities and their context, the task is to predict the label of the relation out of the set of possible classes, namely `USAGE`, `RESULT`, `MODEL-FEATURE`, `PART_WHOLE`, `TOPIC`, `COMPARISON`. The task is decomposed into two different scenarios according to the data used, namely with either manually annotated entities (1.1) or noisy data with automatically extracted entities (1.2).

2. **Relation Extraction and Classification.** This subtask addresses the whole end-to-end pipeline of relation extraction. Given abstracts of scientific papers annotated with entities, systems are required to extract pairs of entities in a semantic relation, as well as assign a label and directionality to the extracted relation.

We participated in the subtask 1.1, relation classification on clean data, and subtask 2 (relation extraction only). Our approach relies on supervised learning using Support Vector Machines (SVMs), k-Nearest Neighbors (kNNs), and Convolutional Neural Networks (CNNs). We were ranked 16th on task 1.1 with an F1 score 44.0% and 7th on task 2 with an F1 of 28.4%.

2 Related Work

A series of supervised systems for extracting keyphrases and relations in scientific publications was presented in the context of the SemEval task 10 in 2017 (Augenstein et al., 2017). In contrast to the present task formulation, the set of relations consisted only of two, namely hyponymy and synonymy. For this task, the best systems were those based on neural approaches.

Lee et al. (2017) obtained the highest score by employing a convolutional neural network with a specific embedding layer encoding manually crafted features such as the word, the position of the word and the part-of-speech (POS) tag. The second best system was a neural end-to-end model by Ammar et al. (2017). It predicted the relation types based on a context-sensitive representation of the keyphrases which they obtained by using a variety of information, e.g., entity type embeddings and syntactic and sequential path information generated by a bidirectional Long Short-Term Memory (LSTM) layer. As opposed to the former systems, the approach of Barik and Marsi (2017), ranked third, was based on manually crafted features and more traditional classifiers such as decision trees and SVMs.

Proceedings of the 12th International Workshop on Semantic Evaluation (SemEval-2018), pages 826–830
New Orleans, Louisiana, June 5–6, 2018.

Outside SemEval, Zelenko et al. (2003) proposed different kernel methods combined with the SVM and Voted Perceptron learning algorithms for extracting `person-affiliation` and `organization-location` relations. Another kernel-based approach using different sources of syntactic information was presented by Zhao and Grishman (2005).

Our neural approach is inspired by the work of Nguyen and Grishman (2015), who employ a CNN with word embedding and position embedding lookup. Embeddings are concatenated for obtaining positionally and semantically sensitive token representations and then fed into a convolutional layer, followed by a maximum pooling layer and a fully connected feed-forward network with a softmax classifier.

3 Methodology

3.1 Feature-based Approach

Feature Engineering. For our feature-based approach we explored the following pool of features.

1. **Structural Features.** We compute for each entity the distance to the last and next entity and the relative position of the entity in the sentence. Furthermore, assuming that verb phrases are a strong indicator for the specific type of relationship, we also add the distance to the next verb.

2. **Lexical features.** We include the first and last five letters of the given surface form of each entity. Furthermore, for each word we add the last and next three words in the surrounding context window and the word length. Again emphasizing the relative importance of verbs, we specifically encode the last and the next verb. Manual exploration of the training data revealed patterns such as the preposition 'of' and the verb 'use' being a strong indicator of the relations `PART_WHOLE` and `USAGE` respectively. Therefore, we add two binary features `hasOf` and `hasUse`. The last lexical feature we employ is the Tf-Idf representation of the sentence.

3. **Syntactic features.** We use the POS tag of each word as shallow syntactic feature.

4. **Semantic features.** In order to add semantic information we employ an embedding representation of the sentence by averaging the GloVe word embeddings pretrained on Wikipedia 2014 and Gigaword 5.[1]

Experimental Setup. We experimented with two classifiers, namely linear SVM and kNN. Both models were wrapped in a 5-fold cross validation in order to obtain performance estimates on the whole training set. Furthermore, we tuned the hyperparameters of the models by nesting the cross validation in a grid search, optimizing the penalty term c of the linear SVM given the search space $c \in \{0, 001, 0.01, 0.1, 1, 10\}$ and weighting the penalty term either balanced according to the distribution of the classes or leaving all classes with equal penalty. In the relation classification, we also experimented with one-vs-rest and Crammer-Singer as multi-class classification strategies.

Similarly, for the kNN we optimized the number of neighbors in the search space $k \in \{3, 5, 7, 9\}$ and tried uniform weighting and weighting neighbors based on their inverse distance. All other parameters were left to the default values provided by scikit-learn,[2] which we used as implementation framework.

To find suitable subsets of our heuristically extracted pool of features, a forward feature selection was employed.

3.2 Deep Learning-based Approach

Model Architecture. We employ a CNN (LeCun and Bengio, 1998) for the relation extraction task, which was first introduced to the natural language processing community by Collobert and Weston (2008). Our CNN architecture is inspired by Nguyen and Grishman (2015) and consists of four main layers: (1) embedding layer, (2) convolutional layer, (3) maximum pooling layer, and (4) fully connected output layer.

In a first step, given a word and its relative position in the sentence, we lookup the accompanying vector representations and concatenate them in order to obtain a position-sensitive semantic representation of the token. Next, the model convolutes, i.e., slides, over the embeddings to capture the context of the token in a window of size k, which is followed by subsampling the obtained matrices using maximum pooling, e.g., preserving the N maximal values. This allows the model to recognize the most informative k-grams for the task. In a last step, the representations are fed into a fully

[1] https://nlp.stanford.edu/projects/glove/

[2] http://scikit-learn.org/

word	d1	d2	entity ID
word	0	-9	C08-1105.1
Semantic	0	-8	C08-1105.1
Role	0	-7	C08-1105.1
Labeling	1	-6	none
are	2	-5	none
usually	3	-4	none
limited	4	-3	none
in	5	-2	none
a	6	-1	none
syntax	7	0	C08-1105.3
subtree	8	0	C08-1105.3

Table 1: Example of how we assign relative position to training and test instances.

connected layer followed by a softmax classifier predicting the label.

Experimental Setup. As input to the model we feed pairs of candidate entities together with the textual content between them. As the CNN expects fixed-size vector representations, we pad the texts to the length of the longest sequence using a special token to which we assign a random embedding vector. We compute the relative position for each word w_i as distance vector $d_i = [d_{i1}, d_{i2}]$. Table 1 shows a example of how we assign relative distances to each word in our instances. The word embedding matrix is initialized with 300-dimensional domain-specific word embeddings from Lauscher et al. (2017), which showed during prototyping superior performance when compared with standard domain-independent ones. Finally, we obtain a position-sensitive representation of the word w_i by concatenating its embedding e_i as well as the position embedding into a single vector $v_i = [e_i, d_{i1}, d_{i2}]$.

Since the data for the relation extraction task is highly skewed towards the number of negative instances (i.e., only 1,228 out of 8,768 are positive), we decided to experiment with data balancing techniques. More specifically, we tried upsampling the positive instances applying an upsampling rate $r_u \in [1, 5]$ and similarly, downsampling the negative instances with a rate $r_d \in [0.1, 1]$.

Optimizing the hyperparameters of our model, we experimented with the numbers of CNN filters in range $f \in [50, 500]$ and with a filter size $s \in [3, 15]$. For the regularization of our model we apply dropout before the fully connected layer and experimented with a dropout keep probability of $d \in [0, 1]$. For all other hyperparameters we use the values from Nguyen and Grishman (2015). Last, in order to make our models comparable among each other, we nested the CNN in a 5-fold cross validation.

Class	Count	Ratio
Usage	483	39.33%
Model-Feature	326	26.55%
Part_whole	234	19.06%
Topic	95	07.74%
Result	72	05.86%
Compare	18	01.47%
Total	**1,228**	**100%**

Table 2: Class distribution in the training data.

4 Evaluation

Here, we briefly give an overview of the data provided by the organizers of the shared task as well as our submitted runs. We also present and discuss the final results achieved.

Data. The training data provided by the task organizers for subtask 1.1 and subtask 2 is composed of 350 abstracts of scientific publications with manually annotated entities and relations. In total, the number of relation instances amounts to 1,228 samples. Table 2 shows the distribution of the labels for the relation classification.

For the relation extraction task, entity pairs for all semantic relations appear in the very same sentence. Therefore, we generate relation candidates by pairing all entity mentions found within the same sentence boundary. This way we end up with 8,386 candidate entity pairs among which 1,228 are positive instances.

The test data for task 1.1 and 2 was provided in the same format as training data, containing 150 abstracts and 355 relation instances (for subtask 1.1 only).

Submitted runs. We selected two models for the relation classification and three models for the relation extraction task for the final submission according to their scores on the development data using the official scoring script. The model configurations are summarized in Table 3 and Table 4, respectively.

Final Results. The official scores for the submitted models are listed in Tables 5 and 6.

Model	Hyperparameter	Choice	Features
SVM	c	1	tfidf
	multi class	one-vs-rest	isReverse
	class weight	balanced	nextEntityDist(x)
			position(y)
			POStag(x)
kNN	k	5	tfidf
	weights	distance	isReverse
			hasOf

Table 3: Submitted models for task 1.1 (relation classification). In the features listed, x represents the first entity while y represents the second entity of a candidate pair.

Model	Hyperparameter	Choice	Features
SVM	c	0.1	position(x,y)
	class weight	balanced	lastEntityDist(y)
			nextEntityDist(x)
			tokenLength(y)
			POStag(x,y)
			firstLetter(y)
kNN	k	3	relativePosition(x,y)
	weights	distance	lastEntityDist(y)
			position(y)
			avgEmbedding
			hasUse
			POStag(x,y)
CNN	dropout	0.5	word embedding
	upsampling rate	3	relative position
	number of filters	200	
	filter size	3-8	

Table 4: Submitted models for task 2 (relation extraction). In the features listed, x represents the first entity while y represents the second entity of a candidate pair.

In the relation classification task, SVM achieves better performance than kNN by a large margin, while in the relation extraction task, it is the deep learning models that perform best. Within the scope of the traditional models, SVM consistently outperforms kNN for both the classification (on every relation type, cf. Table 5), as well as the extraction task (Table 6). Furthermore, for task 1.1, SVM performed better than kNN, in every relation type. The reason could be that the vector-like features we used, such as Tf-Idf and the binary POS-tags, suit better for SVM, while kNN was not able to handle the high-dimensional dataset.

For task 2 the linear SVM model results in a relatively high recall but considerably low precision. This result could be related to the following reasons. First, since the training data is highly skewed towards negative examples as described in subsection 4, more false positive cases are predicted. Second, the data is likely to be nonlinearly distributed. As a result, the linear SVM is not able to perform better even when increasing feature dimensionality. In addition to a better feature engineering, we explored the use of an Radial Basis Function (RBF) kernel and Gradient Boosting Tree to increase the precision without hurting the recall. Another interesting point is that the improvement of precision contributed more to the overall F1-score in this official evaluation method. This can be inferred from the results listed in Table 6, where the CNN has higher F1-score, with a higher precision but much lower recall compared to the SVM.

Class	SVM	kNN
Usage	73.68%	63.04%
Model-Feature	51.70%	43.97%
Part_Whole	45.53%	34.90%
Topic	22.22%	00.00%
Result	37.50%	37.50%
Compare	26.32%	12.12%
Macro F1	44.00%	32.49%

Table 5: Results (F1) on relation classification (task 1.1).

	Precision	Recall	F1
SVM	15.67%	90.19%	26.70%
kNN	9.93%	11.44%	10.63%
CNN	18.84%	57.49%	28.38%

Table 6: Results on relation extraction (task 2).

5 Conclusion

In this paper we have presented our approach to the SemEval 2018 Task 7 on extracting and classifying semantic relations from scientific publications. We experimented with feature-based versus neural models. For the classification task, SVM performed better than kNN, although both show problems in predicting minority classes with few examples. For the extraction task, the deep learning method outperformed SVM by a narrow margin. The overall comparable results across methods seem to indicate that, in the future, more work should turn to devising better features or architectures that are able to capture the nuances of semantic relations in the domain of scientific texts.

Acknowledgements. This work was partly funded by the German Research Foundation (DFG), project number EC 477/5-1 (LOC-DB).

References

Waleed Ammar, Matthew Peters, Chandra Bhagavatula, and Russell Power. 2017. The AI2 system at SemEval-2017 task 10 (ScienceIE): semi-supervised end-to-end entity and relation extraction. In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*. Association for Computational Linguistics, Vancouver, Canada, pages 592–596. http://www.aclweb.org/anthology/S17-2097.

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.

Biswanath Barik and Erwin Marsi. 2017. NTNU-2 at SemEval-2017 task 10: Identifying synonym and hyponym relations among keyphrases in scientific documents. In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*. Association for Computational Linguistics, Vancouver, Canada, pages 965–968. http://www.aclweb.org/anthology/S17-2168.

Ronan Collobert and Jason Weston. 2008. A unified architecture for natural language processing: deep neural networks with multitask learning. In *Machine Learning, Proceedings of the Twenty-Fifth International Conference (ICML 2008), Helsinki, Finland, June 5-9, 2008*. pages 160–167. https://doi.org/10.1145/1390156.1390177.

Kata Gábor, Davide Buscaldi, Anne-Kathrin Schumann, Behrang QasemiZadeh, Hafa Zargayouna, and Thierry Charnois. 2018. Semeval-2018 task 7: Semantic relation extraction and classification in scientific papers. In *Proceedings of the 12th International Workshop on Semantic Evaluation (SemEval-2018)*. Association for Computational Linguistics, New Orleans, LA, USA, page tba.

Anne Lauscher, Goran Glavaš, Simone Paolo Ponzetto, and Kai Eckert. 2017. Investigating convolutional networks and domain-specific embeddings for semantic classification of citations. In *Proceedings of the 6th International Workshop on Mining Scientific Publications*. ACM, New York, NY, USA, pages 24–28. https://doi.org/10.1145/3127526.3127531.

Yann LeCun and Yoshua Bengio. 1998. The handbook of brain theory and neural networks. MIT Press, Cambridge, MA, USA, chapter Convolutional Networks for Images, Speech, and Time Series, pages 255–258.

Ji Young Lee, Franck Dernoncourt, and Peter Szolovits. 2017. MIT at SemEval-2017 task 10: Relation extraction with convolutional neural networks. In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*. Association for Computational Linguistics, Vancouver, Canada, pages 978–984. http://www.aclweb.org/anthology/S17-2171.

Thien Huu Nguyen and Ralph Grishman. 2015. Relation extraction: Perspective from convolutional neural networks. In *Proceedings of the 1st Workshop on Vector Space Modeling for Natural Language Processing*. Association for Computational Linguistics, Denver, Colorado, pages 39–48. http://www.aclweb.org/anthology/W15-1506.

Dmitry Zelenko, Chinatsu Aone, and Anthony Richardella. 2003. Kernel methods for relation extraction. *Journal of Machine Learning Research* 3:1083–1106. http://www.jmlr.org/papers/v3/zelenko03a.html.

Shubin Zhao and Ralph Grishman. 2005. Extracting relations with integrated information using kernel methods. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL'05)*. Association for Computational Linguistics, Ann Arbor, Michigan, pages 419–426. https://doi.org/10.3115/1219840.1219892.