ClaiRE at SemEval-2018 Task 7: Classification of Relations using Embeddings

Lena Hettinger, Alexander Dallmann, Albin Zehe, Thomas Niebler and Andreas Hotho
DMIR Group
University of Wuerzburg
{hettinger,dallmann,zehe,niebler,hotho}
@informatik.uni-wuerzburg.de

Abstract

In this paper we describe our system for SemEval-2018 Task 7 on classification of semantic relations in scientific literature for clean (subtask 1.1) and noisy data (subtask 1.2). We compare two models for classification, a C-LSTM which utilizes only word embeddings and an SVM that also takes hand-crafted features into account. To adapt to the domain of science we train word embeddings on scientific papers collected from arXiv.org. The hand-crafted features consist of lexical features to model the semantic relations as well as the entities between which the relation holds. **Cla**ssi**f**ication of **R**elations using **E**mbeddings (ClaiRE) achieved an F1 score of 74.89 % for the first subtask and 78.39 % for the second.

1 Introduction

The goal of SemEval-2018 Task 7 is to extract and classify semantic relations between entities into six categories that are specific to scientific literature (Gábor et al., 2018). In this work, we focus on the subtask of classifying relations between entities in manually (subtask 1.1) and automatically annotated and therefore noisy data (subtask 1.2). Given a pair of related entities, the task is to classify the type of their relation among the following options: `Compare`, `Model-Feature`, `Part_Whole`, `Result`, `Topic` or `Usage`. Relation types are explained in detail in the task description paper (Gábor et al., 2018). The following sentence shows an example of a `Result` relation between the two entities **combination methods** and **system performance**:

> **Combination methods** are an effective way of improving **system performance**.

This sentence is a good example for two challenges we face in this task. First, almost half of all entities consist of noun phrases which has to be considered when constructing features. Secondly, the vocabulary is domain dependent and therefore background knowledge should be adopted.

Previous approaches for semantic relation classification tasks mainly employed two strategies. Either they made use of a lot of hand-crafted features or they utilized a neural network with as few background knowledge as possible. The winning system of an earlier SemEval challenge on relation classification (Hendrickx et al., 2009) adopted the first approach and achieved an F1 score of 82.2% (Rink and Harabagiu, 2010). Later, other works outperformed this approach by using CNNs with and without hand-crafted features (Santos et al., 2015; Xu et al., 2015) as well as RNNs (Miwa and Bansal, 2016).

Approach We present two approaches that use different levels of preliminary information. Our first approach is inspired by the winning method of the SemEval-2010 challenge (Rink and Harabagiu, 2010). It models semantic relations by describing the two *entities*, between which the semantic relation holds, as well as the words between those entities. We call those in-between words the *context* of the semantic relation. We classify relations by using an SVM on lexical features, such as part-of-speech tags. Additionally we make use of semantic background knowledge and add pre-trained word embeddings to the SVM, as word embeddings have been shown to improve performance in a series of NLP tasks, such as sentiment analysis (Kim, 2014), question answering (Chen et al., 2017) or relation extraction (Dligach et al., 2017). Besides using existing word embeddings generated from a general corpus, we also train embeddings on scientific articles that better reflect scientific vocabulary.

In contrast, our second approach relies on word

Proceedings of the 12th International Workshop on Semantic Evaluation (SemEval-2018), pages 836–841
New Orleans, Louisiana, June 5–6, 2018.

embeddings only, which are fed into a convolutional long-short term memory (C-LSTM) network, a model that combines convolutional and recurrent neural networks (Zhou et al., 2015). Therefore no hand-crafted features are used. Because both CNN and RNN models have shown good performance for this task, we assume that a combination of them will positively impact classification performance compared to the individual models.

By combining lexical information and domain-adapted scientific word embeddings, our system ClaiRE achieved an F1 score of 74.89% for the first subtask with manually annotated data and 78.39% for the second subtask with automatically annotated data.

2 Features

In this section, we describe the features which are used in our two approaches. All sentences are first preprocessed before constructing boolean lexical features on the one hand and word embedding vectors on the other. Both feature groups are based on the entities of relations as well as the context in which those entities appear.

Apart from the `Compare` relation, all relation types are asymmetric, and therefore the distinction between start and end entity of a relation is important. If entities appear in reverse order, that means the end entity of a relation appears first in the sentence, this is marked by a *direction* feature which is part of the data set.

In our entrance example, **combination methods** denotes the start entity, **system performance** the end entity, and **are an effective way of improving** the context.

2.1 Preprocessing

Early experiments showed that it is beneficial to filter the vocabulary of our data and reduce noise by leaving out infrequent context words. The best setting was found to be a frequency threshold of 5 on lemmatized words. Therefore we discard a context word if its lemma appears less than 5 times in the corpus of the respective subtask.

2.2 Context features

First we will explain feature construction based on the context of a relation. Abbreviations for feature names are denoted in brackets. Context is defined as the words between two entities. Early tests showed that using those words described the relation better than the words surrounding the relation entities.

Lexical We construct several lexical boolean features which are illustrated in Table 1. First we apply a bag of words (bow) approach where each lemmatized word forms one boolean feature, which for example takes 1 as value if the lemma *improve* is present and 0 if it is not. Second we determine whether the context words contain certain part-of-speech (POS) tags (pos), such as *VERB*. POS-tagging was done with the help of SpaCy[1] (`v.2.0.2`). To represent the structure of the context phrase we add a path of POS tags feature, which contains the order in which POS tags appear ($pospath$). The distance feature depicts whether the POS-path and therefore the context phrase has a certain length ($dist$).

Additionally we add background knowledge by extracting the top-level Levin classes of intermediary verbs from VerbNet[2] (lc), a verb lexicon compatible with WordNet. It contains explicitly stated syntactic and semantic information, using Levin verb classes to systematically construct lexical entries (Schuler, 2005). For example the verb *improve* belongs to class 45.4, which is described by Levin as consisting of "alternating change of state" verbs.[3]

Embeddings Aside from lexical features we also use word embedding vectors to leverage information from the context of entities (c). For each filtered context word we extract its word embeddding from a pre-trained corpus, where out-of-vocabulary words (OOV) are represented by the zero vector. The individual word vectors are later applied to train a C-LSTM.

In contrast, for use in an SVM we found it beneficial to represent the context embedding features as the average over all context word embeddings.

2.3 Entity features

In the second set of features, we model the relation entities themselves as they may be connected to a certain relation class. For example, the token *performance* or one form of it mostly appears as an end entity of a `Result` relation, and in the rare

[1] https://spacy.io/

[2] http://verbs.colorado.edu/~mpalmer/projects/verbnet.html

[3] http://www-personal.umich.edu/~jlawler/levin.verbs

Example Sentence: Combination methods are an effective way of improving **system performance**.

Lexical Feature Set	Exemplary Boolean Features
BagOfWords (bow)	an, be, effective, improve, of, way
POS tags (pos)	ADJ, ADP, DET, NOUN, VERB
POS path ($pospath$)	VDANAV
Distance ($dist$)	6
Levin classes (lc)	45
Entities without order ($ents$)	combination methods, methods, system performance, performance
Start entity ($startEnt$)	combination methods, methods
End entitiy ($endEnt$)	system performance, performance
Similarity ($sim100$)	0.43
Similarity bucket ($simb$)	q50

Table 1: Examples for lexical context and entity features.

case when it represents a start entity, it is almost always part of a `Compare` relation. Therefore we leverage information about entity position for the creation of lexical and embedding entity features.

Lexical For the creation of boolean lexical features, we first take the lowercased string of each entity and construct up to three distinct features from it. One feature which marks its general appearance in the corpus without order ($ents$) and one each if it occurs as start ($startEnt$) or end ($endEnt$) entity of a relation, taking its direction into account. Additionally we add the head noun to the respective feature set if the entity consists of a nominal phrase to create greater overlap between instances.

Furthermore we measure the semantic similarity of the relation entities using the cosine of the corresponding word embedding vectors ($sim100$). While the cosine takes every value from [-1, 1] in theory, we cut off after two digits to reduce the feature space and get 99 boolean similarity features for our corpus. To again enable learning across instances we additionally discretize the similarity range and form another five boolean similarity features ($simb$) that capture into which of the following buckets the similarity score falls: $q0 = [-1, 0), q25 = [0, 0.25), q50 = [0.25, 0.5), q75 = [0.5, 0.75), q100 = [0.75, 1]$ (values below zero are very rare in this corpus).

Embeddings Similar to the context features we also want to add word embeddings of entities to our entity feature set. This is not straighforward as more than 44% of all entities consist of nominal phrases, while a word embedding usually corresponds to a single word. By way of comparison, the proportion of nominals in the relation classification corpus of the SemEval-2010 challenge was only 5%. Thus we tested different strategies to obtain a word embedding for nominal phrases and found that averaging over the individual word vectors of the phrase yielded the best results for this task. These word embeddings for start (e_s) and end (e_e) entities of relations were then presented to our two classification methods, which will be described in detail in the following section.

3 Classification Methods

We utilize two different models for classifying semantic relations: an SVM which incorporates both the lexical and embedding features described in Section 2 and a Convolutional Long Short Term Memory (C-LSTM) neural network that only uses word embedding vectors

To fully exploit our hand-crafted lexical features we employ a traditional classifier. In comparison to Naive Bayes, Decision Trees and Random Forests we found a Support Vector Machine to perform best for this task. Instead of utilizing the decision function of the SVM to predict test labels we decided to make use of the probability estimates according to Wu et al. (2004) as this proved to be more successful. As mentioned before, the lexical features are fed into the SVM as boolean features whereas the word embeddings are normalized using MinMax-Scaling to the range $[0, 1]$ to make it easier for the SVM to handle both feature groups.

In contrast to SVM, neural network models do not necessarily rely on handcrafted features and are therefore faster to implement. We experiment with standard C-LSTM (Zhou et al., 2015) which extracts a sentence representation by combining one-dimensional convolution and an LSTM network and uses the representation to perform a classification.

label	subtask 1.1	subtask 1.2	total
COMPARE	95 (8%)	41 (3%)	136 (5%)
MODEL-F.	326 (27%)	174 (14%)	500 (20%)
PART_W.	234 (19%)	196 (16%)	430 (17%)
RESULT	72 (6%)	123 (10%)	195 (8%)
TOPIC	18 (1%)	243 (20%)	261 (11%)
USAGE	483 (39%)	468 (38%)	951 (38%)

Table 2: Distribution of class labels for training data as absolute and relative values.

4 Evaluation

After describing the two models we employ for relation classification, we now portray the data set we use and present results for both SVM and C-LSTM as micro-F1 and macro-F1. The latter is the official evaluation score of the SemEval Challenge. We describe the experimental setup for both models and compare different feature sets and pre-trained embeddings.

4.1 Data and Background Knowledge

We evaluate our approach on a set of scientific abstracts, D_{test}. It consists of 355 semantic relations for each subtask which are similarly distributed as its respective training data set. As training data we received 350 abstracts of scientific articles per subtask, which resulted in 1228 labeled training relations for subtask 1.1 and 1245 training instances for subtask 1.2 (c.f. Table 2). We combine data sets of both subtasks for training, resulting in 2473 training examples in total (D_{train}).

Background Knowledge In our experiments, we compare different pre-trained word embeddings as a source of background knowledge. As a baseline, we employ a publicly available set of 300-dimensional word embeddings trained with GloVe (Pennington et al., 2014) on the Common Crawl data[4] (*CC*). To better reflect the semantics of scientific language, we trained our own scientific embeddings using word2vec (Mikolov et al., 2013) on a large corpus of papers collected from arXiv.org[5] (*arXiv*).

In order to create the scientific embeddings, we downloaded LaTeX sources for all papers published in 2016 on arXiv.org using the provided dumps.[6] After originally trying to extract the plain text from the sources, we found that it was more feasible to first compile the sources to pdf (excluding all graphics etc.) and then use pdftotext[7] to convert the documents to plain text. This resulted in a dataset of about 166 000 papers. Using gensim (Řehůřek and Sojka, 2010), for each document we extracted tokens of minimum length 1 with the wikicorpus tokenizer and used word2vec to train 300-dimensional word embeddings on the data. We kept most hyper-parameters at their default values, but limited the vocabulary to words occurring at least 100 times in the dataset, reducing for example the noise introduced by artifacts from equations.

	context		+ entities	
data	macro F1	micro F1	macro F1	micro F1
1.1	44.35	58.87	50.30	65.63
+1.2	47.43	61.69	64.38	69.30
CC	51.79	65.07	72.47	74.93
arXiv	52.15	65.92	**74.89**	**76.90**

Table 3: SVM results for subtask 1.1.

	context		+ entities	
data	macro F1	micro F1	macro F1	micro F1
1.2	67.25	69.75	72.48	80.39
+1.1	64.54	69.30	74.69	83.10
CC	62.64	70.70	75.87	**84.79**
arXiv	63.07	70.70	**78.39**	83.10

Table 4: SVM results for subtask 1.2.

4.2 Parameters and Results

After an extensive grid search per cross validation the best parameters for the SVM were found to be a rbf-kernel with $C = 100$ and $\gamma = 0.001$ for both tasks.

Results of the SVM for subtask 1.1. are shown in Table 3. Adding entity features proves to be very beneficial compared to using only context features, as we could improve macro-F1 by 12 points on average. Results are further improved by enlarging the data set with the training samples of subtask 1.2 and by adding word embeddings to the feature set. While adding the CC embeddings enhances the micro-F1 by more than 4 points, our domain-adapted arXiv embeddings prove to perform even better and deliver the best result with a macro-F1 score of 74.89 % and a micro-F1 of 76.90 %.

Similar observations can be made for subtask 1.2., as is pictured in Table 4.

Due to space limitations we publish parameter

[4] http://commoncrawl.org/
[5] https://arxiv.org
[6] https://arxiv.org/help/bulk_data
[7] https://poppler.freedesktop.org

details and elaborate results for the C-LSTM on arXiv.org (Hettinger et al., 2018). In comparison to the SVM, which additionally uses hand-crafted features, the C-LSTM achieves lower scores. For arXiv embeddings it reaches a macro-F1 of 63.3 % for the first subtask and 68.0 % for the second.

5 Discussion

We briefly discuss our approach during the training phase of the SemEval-Challenge and how label distribution and evaluation measure influences our results. Ahead of the final evaluation phase where the concealed test data D_{test} was presented to the participants we were given a preliminary test partition D_{pre} as part of the training data D_{train}. To be able to estimate our performance we evaluated it on D_{pre} as well as for a 10-fold stratified cross validation setting. We chose this procedure to be sure to pick the best system for submission at the challenge.

As some classes were strongly underrepresented in the training corpus and D_{pre}, we assumed that this is also true for the final test set D_{test}. When in doubt we therefore chose to optimize according to D_{pre} as cross validation is based on a slightly more balanced data set (of train data for subtask 1.1 + 1.2). The best system we submitted for subtask 1.1 of the challenge achieved a macro-F1 of 75.05% on D_{pre} during the training phase which shows that we were able to estimate our final result pretty closely.

During training we also noticed that for heavily skewed class distributions as in this case, macro-F1 as an evaluation measure strongly depends on a good prediction of very small classes. For example, macro-F1 of subtask 1.1 increases by 5 points if we correctly predict one `Topic` instance out of three instead of none. Thus we pick a configuration that optimizes the small classes.

We also omitted some lexical feature sets from our system as performance on the temporary and final test set showed that they did not improve results. These features were hypernyms of context and entity tokens from WordNet and dependency paths between entities. Using tf-idf normalization instead of boolean for lexical features also worsened our results.

6 Conclusion

In this paper, we described our SemEval-2018 Task 7 system to classify semantic relations in scientific literature for clean (subtask 1.1) and noisy (subtask 1.2) data. We constructed features based on relation entities and their context by means of hand-crafted lexical features as well as word embeddings. To better adapt to the scientific domain, we trained scientific word embeddings on a large corpus of scientific papers obtained from arXiv.org. We used an SVM to classify relations and additionally contrasted these results with those obtained from training a C-LSTM model on the scientific embeddings. We were able to obtain a macro-F1 score of 74.89 % on clean data and rank 4th out of 28 and 78.39 % on noisy data, which resulted in a 6th place out of 20.

In future work, we will improve the tokenization of the scientific word embeddings and also take noun compounds into account, as they make up a large part of the scientific vocabulary. We will also investigate more complex neural network based models, that can leverage additional information, for example relation direction and POS tags. Some minor changes we applied to the feature generation during the post-evaluation phase and which further improved our results by more than 2 % are published on arXiv.org together with more detailed evaluation (Hettinger et al., 2018).

References

Danqi Chen, Adam Fisch, Jason Weston, and Antoine Bordes. 2017. Reading wikipedia to answer open-domain questions. In *ACL (1)*, pages 1870–1879. Association for Computational Linguistics.

Dmitriy Dligach, Timothy Miller, Chen Lin, Steven Bethard, and Guergana Savova. 2017. Neural temporal relation extraction. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*, volume 2, pages 746–751.

Kata Gábor, Davide Buscaldi, Anne-Kathrin Schumann, Behrang QasemiZadeh, Haïfa Zargayouna, and Thierry Charnois. 2018. Semeval-2018 Task 7: Semantic relation extraction and classification in scientific papers. In *Proceedings of International Workshop on Semantic Evaluation (SemEval-2018)*, New Orleans, LA, USA.

Iris Hendrickx, Su Nam Kim, Zornitsa Kozareva, Preslav Nakov, Diarmuid Ó Séaghdha, Sebastian Padó, Marco Pennacchiotti, Lorenza Romano, and Stan Szpakowicz. 2009. Semeval-2010 task 8: Multi-way classification of semantic relations between pairs of nominals. In *Proceedings of the Workshop on Semantic Evaluations: Recent Achievements and Future Directions*, pages 94–99. Association for Computational Linguistics.

Lena Hettinger, Alexander Dallmann, Albin Zehe, Thomas Niebler, and Andreas Hotho. 2018. Claire at semeval-2018 task 7 - extended version. *Computation and Language Repository*, arXiv:1804.05825.

Yoon Kim. 2014. Convolutional neural networks for sentence classification. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing, EMNLP 2014, October 25-29, 2014, Doha, Qatar, A meeting of SIGDAT, a Special Interest Group of the ACL*, pages 1746–1751.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In *NIPS*, pages 3111–3119. Curran Associates, Inc.

Makoto Miwa and Mohit Bansal. 2016. End-to-end relation extraction using lstms on sequences and tree structures. Cite arxiv:1601.00770Comment: Accepted for publication at the Association for Computational Linguistics (ACL), 2016. 13 pages, 1 figure, 6 tables.

Jeffrey Pennington, Richard Socher, and Christopher D Manning. 2014. Glove: Global vectors for word representation. In *EMNLP*, volume 14, pages 1532–1543.

Radim Řehůřek and Petr Sojka. 2010. Software Framework for Topic Modelling with Large Corpora. In *Proceedings of the LREC 2010 Workshop on New Challenges for NLP Frameworks*, pages 45–50, Valletta, Malta. ELRA. `http://is.muni.cz/publication/884893/en`.

Bryan Rink and Sanda Harabagiu. 2010. Utd: Classifying semantic relations by combining lexical and semantic resources. In *Proceedings of the 5th International Workshop on Semantic Evaluation*, pages 256–259. Association for Computational Linguistics.

Cicero Nogueira dos Santos, Bing Xiang, and Bowen Zhou. 2015. Classifying relations by ranking with convolutional neural networks. *Proceedings of the 7th International Joint Conference on Natural Language Processing [IJCNLP]*.

Karin Kipper Schuler. 2005. *Verbnet: A Broad-coverage, Comprehensive Verb Lexicon*. Ph.D. thesis, University of Pennsylvania, Philadelphia, PA, USA. AAI3179808.

Ting-Fan Wu, Chih-Jen Lin, and Ruby C. Weng. 2004. Probability estimates for multi-class classification by pairwise coupling. *Journal of Machine Learning Research*, 5(Aug):975–1005.

Kun Xu, Yansong Feng, Songfang Huang, and Dongyan Zhao. 2015. Semantic relation classification via convolutional neural networks with simple negative sampling. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing [EMNLP]*, pages 536–540.

Chunting Zhou, Chonglin Sun, Zhiyuan Liu, and Francis C. M. Lau. 2015. A c-lstm neural network for text classification. *CoRR*, abs/1511.08630.