OhioState at SemEval-2018 Task 7: Exploiting Data Augmentation for Relation Classification in Scientific Papers using Piecewise Convolutional Neural Networks

Dushyanta Dhyani
The Ohio State University OH, USA
dhyani.2@osu.edu

Abstract

We describe our system for SemEval-2018 Shared Task on Semantic Relation Extraction and Classification in Scientific Papers where we focus on the Classification task. Our simple piecewise convolution neural encoder performs decently in an end to end manner. A simple inter-task data augmentation significantly boosts the performance of the model. Our best-performing systems stood 8th out of 20 teams on the classification task on noisy data and 12th out of 28 teams on the classification task on clean data.

1 Introduction

Relation extraction (RE) and Classification (RC) is an integral component of information extraction systems which aim to extract all the entity pairs and their relation $\langle e_1, r, e_2 \rangle$ from a given text corpora. An alternate formulation of relation extraction task focuses on identifying if a relation exists between a predefined pair of entities, and if yes classify from a given set of class relations. RE finds applications in a variety of domains, ranging from knowledge base construction to semantic parsing and question answering. However, the applicability of existing efforts in relation extraction to scientific text calls for a quantitative and qualitative analysis which is the aim of this shared task.

2 Related Work

Existing efforts for RE range from traditional strategies (Qian et al., 2008; Bunescu and Mooney, 2006, 2005; Mintz et al., 2009; Riedel et al., 2010) to more recent end to end deep learning based methods (Zeng et al., 2014, 2015; Lin et al., 2016; Wu et al., 2017) that are more suitable in situations where a lot of training data is available. While a majority of efforts in the RE community are specifically focused towards using distantly supervised data and reduce the associated noise, their discussion is not relevant to the current scenario. The most relevant work is that of (Zeng et al., 2014) who demonstrated the efficacy of convolution neural networks for relation classification and (Zeng et al., 2015) who further enhanced the architecture by proposing the piecewise max-pooling strategy.

3 Task Description

The semantic relation extraction and classification in scientific papers task (Gábor et al., 2018) aims at identifying semantic relations expressed by entity pairs in scientific literature. The contest is further divided into three subtasks, where the first two focus on classification on varying nature of data and the third focuses on extraction task. Since our submitted systems focused only on the classification task, we would from here on discuss mostly about the classification sub-tasks.

3.1 Dataset

The data contains titles and abstracts of papers from ACL Anthology Corpus where entity mentions are either manually annotated (Subtask 1.1 and Subtask 2) or heuristically (Subtask 1.2) determined. However, the relations are manually annotated across all subtasks. For the classification scenario, we are provided with relevant entities and the directionality of their relation. There are 6 class labels: USAGE, RESULT, MODEL, PART_WHOLE, TOPIC, COMPARISON. The classes are highly imbalanced in nature as shown in Fig. 2

3.2 Evaluation

For both Task 1.1 and 1.2, given that the classes are imbalanced, macro-f1 score is used as the official evaluation metric and thus the metric we use for hyperparameter tuning. For more details, we

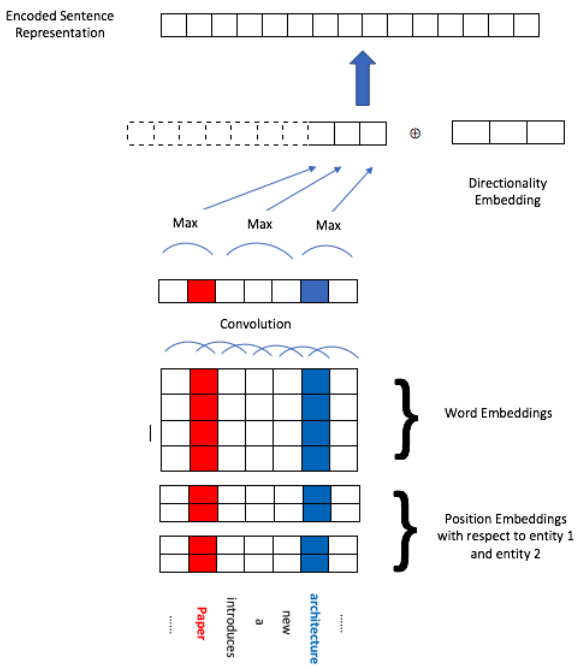
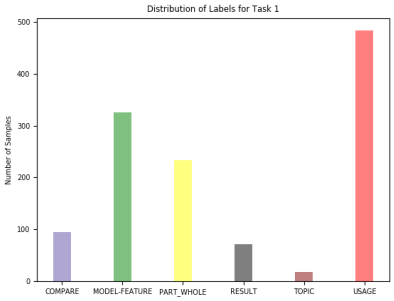
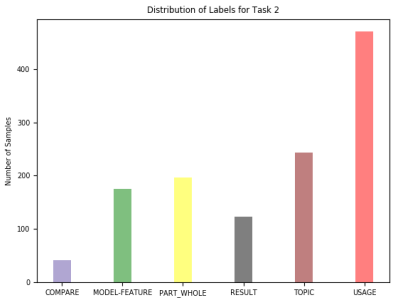


Figure 1: PCNN Encoder with Word, Position and Directionality embeddings



(a) Task 1



(b) Task 2

Figure 2: Class Sizes for Task 1 & Task 2

would refer the reader to the task description paper (Gábor et al., 2018)

4 Methodology

As shown in Fig. 1, we use the piecewise convolutional encoder proposed by (Zeng et al., 2015) which encodes the sentence into an embedding space taking into account the context of text around the entities in an end to end manner. The various components of the encoder are described below

4.1 Preprocessing

Since the original training dataset provided is annotated using XML tags which can be utilized in a variety of ways, we briefly describe our preprocessing steps. Each text item contains a title of a paper and its abstract. Both the entities for a particular training/testing instance could only either be in the title or in the abstract. While it would be interesting to see the impact of incorporating the effect of paper titles on the entities in abstracts and vice versa, to simplify the architecture, we simply treat titles and abstracts as separate and independent sentences. For the representation of entities, the two most obvious options are to either combine sub-words in an entity using a

special character (e.g. *word sense disambiguation* becomes *word_sense_disambiguation*) or to simply use entity head words to represent the starting position of the entity as proposed by (Nguyen and Grishman, 2015). We chose the latter approach for two reasons: 1) The amount of data is relatively small to learn word embeddings on the data itself 2) The conjoined entity representation as in the former approach would probably not exist in the pre-trained word embeddings and thus would have to be replaced by an unknown token. Finally, we used common text cleaning techniques like removing non-alphanumeric characters, replacing all numbers by a unique token, etc.

4.2 Word Representation

Each word in the input is transformed to a static, dense feature representation by looking up a pre-trained word embedding dictionary. We use dependency based word embeddings (Levy and Goldberg, 2014) which incorporate long-range dependencies between words and thus generate embeddings that are more functional in nature (than the traditional bag of words based embeddings) which is presumably more suitable to the current task. All words that do not exist in the dictionary are replaced by *UNK* token and initialized randomly.

4.3 Position Embedding

We also evaluate the distance of each word in the sentence with respect to both *entity 1* and *entity 2* (we limit the values to a maximum distance of *position_window_size*). These position values are then projected into a relatively small embedding space using a trainable embedding layer.

4.4 Directionality Embedding

To incorporate the directionality of the relation exhibited by the two entities ($< e_1, r, e_2 >$ or $< e_2, r, e_1 >$) we also project the direction information into the embedding space by another embedding layer that is trained along with the entire network.

4.5 Convolution and Piecewise Max-Pooling

CNN's have been shown to be good at encoding sentences into vector representations for text classification tasks (Kim, 2014; Zhang et al., 2015; Hu et al., 2014; Kim et al., 2016) and at the same time also speed up the training and inference time. The word representations and position embeddings are concatenated and fed into a convolution encoder which generates features using varying width of filters. To take into account the context of text around and between the entities in consideration, we then perform a piecewise max-pooling operation as shown in Fig. 1. The input representations (word-embedding $\oplus$ position-embedding) are appropriately padded before the convolution operation to ensure that the convolved features have the same length as the input sentence in order to correctly use entity positions for piecewise max-pooling. These features generated by the PCNN are finally concatenated with the directionality embeddings discussed above to generate the sentence level representation.

4.6 Regularization, Output and Training

We use dropout (Srivastava et al., 2014) on the sentence representations with a keep probability of 0.5 as a simple regularization strategy. This is followed by a fully connected layer and a softmax operation for the classification task. We use the standard multi-class cross-entropy loss as our training objective and Adam (Kingma and Ba, 2014) for optimization.

Parameter	Values
Number of Epochs	100,200,400
Maximum Sequence Length	100,200
Batch Size	32,64
Number of Filters	32,64,128
Learning Rate	0.001, 0.0005

Table 1: Hyperparameter Values

5 Experiments

5.1 Data Augmentation

Deep neural models require significant amount of training data to extract relevant features. While our neural model is relatively shallow, the data size for each of the subtask is also small. As a workaround, we simply mix the data from subtask 1.1 with data from subtask 1.2 which hopefully helps in improving the model's generalizability.

5.2 Experimental Settings

While the final training and prediction was performed on the entire training dataset, we use the official validation split provided by contest organizers to perform hyper-parameter tuning. For

Task	Data	Epoch	Batch Size	No. of Filters	Macro-F1 Score
1.1	1.1	200	32	64	35.3
1.1	1.1 + 1.2	200	64	32	**48.1**
1.2	1.2	200	32	64	64.4
1.2	1.1 + 1.2	100	64	128	**74.7**

Table 2: Results of our best performing systems on the official test set with/without data augmentation

the data augmentation scenario, however, we also make use of the validation data from the other task. Given that CNN's are fast to train, we easily use grid search to find the optimal combination of a subset of parameters for each task and each data configuration (with or without augmentation) which are listed in Table 1. For the remaining parameters, we used standard values as recommended by prior literature as follows: convolution filters of width 3,4 and 5; position and directionality embeddings of size 5; windows size for relative positions from entities was set to 30.

6 Results

We report our performance on the classification tasks (Subtask 1.1 and 1.2) according to the official evaluation. While all task settings perform best for a maximum sequence length of 200 and learning rate of 0.001, the rest of the parameters and their corresponding results are listed in Table 2. Even a simple mixing of the two datasets which differ significantly in the nature of tagged entities lead to a significant improvement. Surprisingly though, adding the noisy data to the clean dataset also leads to a 36% increase in performance. This could be attributed to the fact that while heuristically annotated entities are high-level concepts thus sharing a lot of context with similar concepts, most of the manually annotated entities are full noun phrases, thus adding to the complexity of the task. These results also falsify our initial assumption/expectation of Task 1.1 to be easier.

7 Conclusion

We presented a simple end to end model that is fast to train and though does not perform competitively well, makes effective use of additional data for a significant improvement in performance. These results show the effectiveness of mixing/transferring supervision from data coming from a different distribution and thus invites further exploration in semi-supervised/supervised domain adaptation scenarios.

References

Razvan Bunescu and Raymond Mooney. 2005. A shortest path dependency kernel for relation extraction. In *Proceedings of Human Language Technology Conference and Conference on Empirical Methods in Natural Language Processing*, pages 724–731, Vancouver, British Columbia, Canada. Association for Computational Linguistics.

Razvan Bunescu and Raymond J. Mooney. 2006. Subsequence kernels for relation extraction. In *Advances in Neural Information Processing Systems, Vol. 18: Proceedings of the 2005 Conference (NIPS)*.

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.

Baotian Hu, Zhengdong Lu, Hang Li, and Qingcai Chen. 2014. Convolutional neural network architectures for matching natural language sentences. In Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 27*, pages 2042–2050. Curran Associates, Inc.

Yoon 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, Doha, Qatar. Association for Computational Linguistics.

Yoon Kim, Yacine Jernite, David Sontag, and Alexander M Rush. 2016. Character-aware neural language models. In *AAAI*, pages 2741–2749.

Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Omer Levy and Yoav Goldberg. 2014. Dependency-based word embeddings. In *Proceedings of the*

52nd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers), pages 302–308, Baltimore, Maryland. Association for Computational Linguistics.

Yankai Lin, Shiqi Shen, Zhiyuan Liu, Huanbo Luan, and Maosong Sun. 2016. Neural relation extraction with selective attention over instances. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2124–2133, Berlin, Germany. Association for Computational Linguistics.

Mike Mintz, Steven Bills, Rion Snow, and Daniel Jurafsky. 2009. Distant supervision for relation extraction without labeled data. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP*, pages 1003–1011, Suntec, Singapore. Association for Computational Linguistics.

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*, pages 39–48.

Longhua Qian, Guodong Zhou, Fang Kong, Qiaoming Zhu, and Peide Qian. 2008. Exploiting constituent dependencies for tree kernel-based semantic relation extraction. In *Proceedings of the 22nd International Conference on Computational Linguistics (Coling 2008)*, pages 697–704, Manchester, UK. Coling 2008 Organizing Committee.

Sebastian Riedel, Limin Yao, and Andrew McCallum. 2010. Modeling relations and their mentions without labeled text. In *Proceedings of the European Conference on Machine Learning and Knowledge Discovery in Databases (ECML PKDD)*.

Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: A simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*, 15:1929–1958.

Yi Wu, David Bamman, and Stuart Russell. 2017. Adversarial training for relation extraction. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 1778–1783, Copenhagen, Denmark. Association for Computational Linguistics.

Daojian Zeng, Kang Liu, Yubo Chen, and Jun Zhao. 2015. Distant supervision for relation extraction via piecewise convolutional neural networks. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1753–1762, Lisbon, Portugal. Association for Computational Linguistics.

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

Xiang Zhang, Junbo Zhao, and Yann LeCun. 2015. Character-level convolutional networks for text classification. In C. Cortes, N. D. Lawrence, D. D. Lee, M. Sugiyama, and R. Garnett, editors, *Advances in Neural Information Processing Systems 28*, pages 649–657. Curran Associates, Inc.