Texterra at SemEval-2018 Task 7: Exploiting Syntactic Information for Relation Extraction and Classification in Scientific Papers

Andrey Sysoev and **Vladimir Mayorov**
Ivannikov Institute for System Programming of the Russian Academy of Sciences
25 Alexander Solzhenitsyn Street
Moscow, Russia
sysoev@ispras.ru, vmayorov@ispras.ru

Abstract

In this work we evaluate applicability of entity pair models and neural network architectures for relation extraction and classification in scientific papers at SemEval-2018. We carry out experiments with representing entity pairs through sentence tokens and through shortest path in dependency tree, comparing approaches based on convolutional and recurrent neural networks. With convolutional network applied to shortest path in dependency tree we managed to be ranked eighth in subtask 1.1 (”clean data”), ninth in 1.2 (”noisy data”). Similar model applied to separate parts of the shortest path was mounted to ninth (extraction track) and seventh (classification track) positions in subtask 2 ranking.

1 Introduction

Information extraction is an important part of natural language processing. During SemEval-2018 an evaluation devoted to extraction and classification of relations in scientific papers was held (Gábor et al., 2018). The task is described as follows: given abstracts of scientific articles with detected entities, the goal is to choose correct relations for provided $\langle source, target\rangle$ entity pairs (subtask 1 - relation classification) and to determine correct relations among all entity pairs (subtask 2 - relation extraction and classification). The target quality metric in classification is macro-average of F1-scores of every class; for extraction scenario the target metric is F1-score.

Our method is based on multinomial classification of entity pairs and their sentences with neural networks. We experiment with representing entity pairs through all sentence tokens and through tokens along the shortest path between entities in dependency tree (Bunescu and Mooney, 2005). We employ convolutional (CNN) (LeCun et al., 1989) and bidirectional Long Short-Term Memory (biLSTM) (Hochreiter and Schmidhuber, 1997; Tan et al., 2015) neural network based approaches to encode sentences and dependency tree paths.

In this work we mainly focus on relation classification, so most of analysis and experiments are carried out for this task. Slightly modified models which achieve the best results on subtask 1 are adapted for solving relation extraction and classification problem.

The rest of the paper is organized as follows: in section 2 we describe some known approaches for relation extraction and classification. In section 3 our approach is presented in details. Section 4 outlines results of described approach evaluation on official SemEval-2018 task 7 test set. We wrap up with some final thoughts in section 5.

2 Related Work

The relation extraction and classification problem has a long history. Early approaches were based on manually constructed patterns, used to detect entities in the relation under consideration (Blaschke and Valencia, 2001). Further approaches utilized machine learning algorithms (Zelenko et al., 2003) with various hand-crafted features (GuoDong et al., 2005) - syntactic labels, part of speech tags, morphological properties and so on. A brief overview of such methods is presented in (Bach and Badaskar, 2007). Significant part of recent approaches is based on neural networks, trying to eliminate dependency on natural language processing tools: (dos Santos et al., 2015; Lin et al., 2016) use CNNs to extract and classify relations; (Zeng et al., 2014) adapts deep CNNs for the same task. (Socher et al., 2012) introduces recursive neural networks which capture information from phrases and sentences and applies it to relation classification task.

In this work we try to inject extra - syntactic - information gained from natural language processing tools into neural networks based approaches.

3 System Description

Our method is based on multinomial classification with neural networks. The decision about the relation being held is made by analysing sentence, which contains examined entities. Each sentence is represented as a sequence of tokens or as a dependency tree.

3.1 Modelling Tokens

Tokens are encoded with fasttext (Bojanowski et al., 2017). Each token embedding also contains binary indicators of its belonging to source or target entity or other part of the sentence.

3.2 Modelling Entity Pair

The basic way to model entity pair is just to take all tokens of the sentence containing these entities and to encode them with described embedding (section 3.1). Binary indicators of token belonging to entities allow us to distinguish several relations in a single sentence.

Another idea is to use path from source to target entity tokens in dependency tree. In our approach the shortest path is considered: it rises up from source entity directly to the lowest common ancestor and then immediately goes down to target entity tokens (Figure 1). Note that dependency trees are built automatically and are sometimes inconsistent with layout of entities, which may be represented differently in the tree.

When using shortest path in dependency tree, each token embedding is extended with additional information - fasttext of the parent token, syntactic label and direction indicator (whether token is on path from source or target entity to lowest common ancestor).

3.3 Neural Network Architectures

General architecture can be described as follows: some method is utilized to encode sequence of input embeddings into a vector, which is then passed through fully-connected layer L and finally fed into softmax to output predicted label. We experiment with two well-known approaches to encode sequences of input embeddings - biLSTM and CNN.

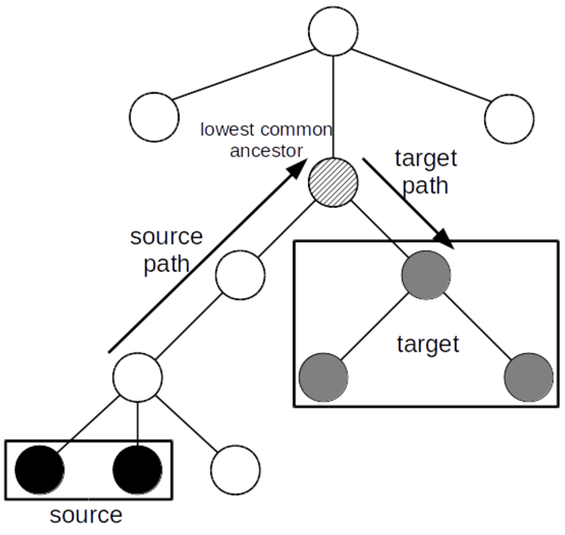

Figure 1: Modelling $\langle source, target\rangle$ entity pair in dependency tree with shortest path.

3.3.1 BiLSTM

BiLSTM-based method is hugely inspired by (Yang et al., 2016). For each sequence item w_k^0 we analyse its nearest context - two items to the left (w_k^{-2}, w_k^{-1}) and to the right (w_k^1, w_k^2). Instead of using w_k^0 directly, its "attentioned" version ω_k is used:

$$\omega_k = \sum_{i=-2}^{2} \alpha_k^i w_k^i, \quad \alpha_k^i = tanh(Ww_k^i + b)^T u,$$

where w_k^i are D-dimensional embedding vectors; b and u are A-dimensional attention vectors; W is an $A \times D$-dimensional attention matrix.

Computed ω_k are further fed into biLSTM network (hidden layer size B). Its final cells output and hidden state together with attention vector (computed similarly to what has been described earlier, but on all biLSTM outputs) are concatenated to form final sequence coding vector.

3.3.2 CNN

Another method is based on CNN. All input sequences are trimmed or padded to fit the same size. Then a number of filters F are applied. Each filter application yields a vector of dimensionality $sequence\_length - filter\_height + 1$; a single maximum value is pulled from each such vector. These values are finally concatenated to form final sequence encoding.

3.3.3 Separate CNNs for Shortest Path Parts

The final method is a modification of CNN one, which is specifically designed to be used when modelling entity pairs with shortest paths. Instead of merging different parts of the path into a single sequence, we use four individual sequences by analogy with (Zeng et al., 2015) - source entity tokens, tokens on path from source to lowest common ancestor, tokens on path from lowest common ancestor to target and target entity tokens - and four separate CNNs for them (sCNN). Outputs of all networks are merged into a single final vector.

3.4 Relation Extraction

We adapt classification approach to relation extraction subtask. The first idea is to apply the same model for seven-class classification (six known relations and absence of relation). Secondly we try two-step approach with successive classifiers of the same architecture: extraction classifier detects entity pairs associated with any relation and then another classifier assigns relation labels for extracted pairs of entities.

For negative examples generation the following strategies are examined: *reflection* - reversed correct asymmetric (all except $COMPARE$) relations are supposed to be negative examples; *in-sentence* - some random portion of entity pairs which co-occur in the same sentence is treated as negative examples.

Finally an attempt to filter out excess relations is made (according to guidelines each entity is allowed to participate in not more than a single relation). We employ greedy method that chooses the most confident relation being held using classifier output weights.

4 Evaluation

We took part in both relation classification and relation extraction subtasks. All results reported in this section are gained on official SemEval-2018 task 7 test data developed by organizers and released after the evaluation phase. Official scores for corresponding submissions are specified in Tables 2 and 3 after the slash sign. The difference is explained by minor parameter variations, typically randomness in variables initialization and number of training epochs.

Relation classification (subtask 1) has two datasets - with manually annotated entities (subtask 1.1 - ”clean data”) and with automatically detected entities (subtask 1.2 - ”noisy data”). We decided to construct a single model merging both datasets into one in order to increase the amount of training examples and to diminish skew in number of sample relations for different types (Table 1).

Relation	**1.1**	**1.2**	**1.1+1.2**
COMPARE	95	41	136
MODEL-FEATURE	326	175	501
PART_WHOLE	234	196	430
RESULT	72	123	195
TOPIC	18	243	261
USAGE	483	470	953

Table 1: Number of relations in subtask datasets.

To encode tokens fasttext (skipgram; minimum length of character n-gram is 1, maximum - 5) is used. We build two separate models with different embedding dimensions - 100 and 300 - using the English Wikipedia.

Evaluation results are presented in Table 2. The target metric is F1. The first part of method name specifies whether all sentence tokens or tokens from shortest path in dependency tree are used. The second part specifies neural network architecture being utilized. We report results for the following neural network parameter values: attention size A is 400; biLSTM hidden layer size B is 1000; CNN filters F - 200 with height 3, 50 with heights 2 and 4, width matches the embedding dimensionality; size of fully-connected layer L is 1000 for biLSTM and 900 for CNN. Specified values are selected during experiments, which are out of this paper scope.

As for subtask 1.1, we conclude that: context attention tends to be beneficial (the only counterexample is sentence biLSTM with fasttext size 300); larger token embeddings are typically better (the only counterexample is sentence biLSTM); syntactic information is helpful for relation classification with neural networks.

For subtask 1.2 the results are more controversial: smaller embeddings sometimes surpass larger ones; utilizing syntactic information seems still beneficial, but the results are not as convincing as in 1.1; in contrast to subtask 1.1 context attention does not tend to improve quality of the approach. From our point of view, such strange behaviour on subtask 1.2 dataset requires further investigation.

Quality evaluations for subtask 2 solutions are presented in Table 3. Target metrics are extrac-

Method	Context attention	Fasttext size	1.1			1.2		
			P	R	F1	P	R	F1
sentence biLSTM	-	100	47.29	57.07	51.72	63.78	60.73	62.21
sentence biLSTM	-	300	52.88	52.21	52.54	73.55	74.92	74.23
sentence biLSTM	+	100	51.14	56.36	53.62	68.17	74.77	71.32
sentence biLSTM	+	300	49.45	55.37	52.24	65.25	67.11	66.17
sentence CNN	-	100	53.53	54.10	53.81	71.62	67.71	69.61
sentence CNN	-	300	57.16	54.96	56.04	73.81	**75.35**	74.57
syntax biLSTM	-	100	55.31	61.71	58.33	71.74	75.13	73.39
syntax biLSTM	-	300	62.35	61.57	61.95	73.76	74.43	74.10
syntax biLSTM	+	100	55.29	**70.97**	62.16	69.52	72.60	71.02
syntax biLSTM	+	300	58.15	70.42	**63.70** / 55.8	63.75	68.62	66.10 / 69.0
syntax CNN	-	100	50.42	61.66	55.48	62.80	65.51	64.13
syntax CNN	-	300	56.27	58.62	57.42 / 64.9	71.79	72.74	72.26 / 74.4
syntax sCNN	-	100	62.83	59.82	61.29	**86.49**	72.87	**79.10**
syntax sCNN	-	300	**64.21**	63.12	63.66 / 62.4	74.22	75.27	74.74 / 73.7

Table 2: Final quality results for subtasks 1.1 and 1.2.

Method	Negative examples	Extraction			Classification		
		P	R	F1	P	R	F1
7-class biLSTM	reflections	8.11	81.74	14.76	4.43	53.58	8.19
7-class CNN	reflections	8.37	82.83	15.21	5.26	56.31	9.63
7-class sCNN	reflections	7.94	80.11	14.44 / 15.6	6.12	66.08	11.21 / 9.6
2-step biLSTM	reflections	7.77	92.64	14.33 / 14.4	3.49	53.35	6.55 / 8.0
2-step CNN	reflections	7.98	88.56	14.63 / 13.9	4.64	52.11	8.53 / 8.2
2-step sCNN	reflections	7.97	79.84	14.50	4.81	52.03	8.80
7-class sCNN	reflections + 20%	8.20	83.65	14.94	8.05	57.71	14.13
7-class sCNN	reflections + 50%	8.55	67.85	15.19	6.22	57.78	11.23
7-class sCNN	reflections + 100%	8.43	**93.19**	15.46	5.37	**66.13**	9.94
2-step sCNN	reflections + 20%	10.36	81.47	18.38	6.47	53.24	11.55
2-step sCNN	reflections + 50%	14.67	65.94	**24.00**	9.53	41.25	15.49
2-step sCNN	reflections + 100%	11.40	81.20	19.99	6.66	52.17	11.81
7-class sCNN + filter	reflections + 20%	14.73	43.05	21.94	13.69	36.54	19.91
2-step sCNN + filter	reflections + 50%	**20.36**	28.07	23.60	**17.25**	26.47	**20.89**

Table 3: Final quality results for subtask 2.

tion and evaluation F1. All experiments are performed with the same input vector size (fasttext dimensionality equals to 300); entity pairs are modelled with shortest path in dependency tree. The second column specifies negative examples generation strategy: reflection with some portion (0-100%) of in-sentence negative examples is always used. For training purposes both subtask 1.1 and 1.2 data is utilized.

When reflection strategy for negative examples generation is used seven-class approach performs better. With utilization of both strategies two-step approach breaks forward. Post-processing improves quality for both approaches, however it is still rather low compared with the results of other participants.

5 Conclusion

In this work we tried to study how utilization of syntactic information influences the quality of relation extraction and classification in scientific papers. According to our experiments the approach based on shortest path in dependency tree yields the best results. The actual network architecture delivering the best result depends on the subtask being solved.

References

Nguyen Bach and Sameer Badaskar. 2007. A review of relation extraction. *Literature review for Language and Statistics II*, 2.

Christian Blaschke and Alfonso Valencia. 2001. Can bibliographic pointers for known biological data be found automatically? protein interactions as a case study. *Comparative and Functional Genomics*, 2(4):196–206.

Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. 2017. Enriching word vectors with subword information. *Transactions of the Association for Computational Linguistics*, 5:135–146.

Razvan C Bunescu and Raymond J Mooney. 2005. A shortest path dependency kernel for relation extraction. In *Proceedings of the conference on human language technology and empirical methods in natural language processing*, pages 724–731. Association for Computational Linguistics.

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.

Zhou GuoDong, Su Jian, Zhang Jie, and Zhang Min. 2005. Exploring various knowledge in relation extraction. In *Proceedings of the 43rd annual meeting on association for computational linguistics*, pages 427–434. Association for Computational Linguistics.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation*, 9(8):1735–1780.

Yann LeCun, Bernhard Boser, John S Denker, Donnie Henderson, Richard E Howard, Wayne Hubbard, and Lawrence D Jackel. 1989. Backpropagation applied to handwritten zip code recognition. *Neural computation*, 1(4):541–551.

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)*, volume 1, pages 2124–2133.

Cicero dos Santos, Bing Xiang, and Bowen Zhou. 2015. Classifying relations by ranking with convolutional neural networks. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, volume 1, pages 626–634.

Richard Socher, Brody Huval, Christopher D Manning, and Andrew Y Ng. 2012. Semantic compositionality through recursive matrix-vector spaces. In *Proceedings of the 2012 joint conference on empirical methods in natural language processing and computational natural language learning*, pages 1201–1211. Association for Computational Linguistics.

Ming Tan, Cicero dos Santos, Bing Xiang, and Bowen Zhou. 2015. Lstm-based deep learning models for non-factoid answer selection. *arXiv preprint arXiv:1511.04108*.

Zichao Yang, Diyi Yang, Chris Dyer, Xiaodong He, Alex Smola, and Eduard Hovy. 2016. Hierarchical attention networks for document classification. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1480–1489.

Dmitry Zelenko, Chinatsu Aone, and Anthony Richardella. 2003. Kernel methods for relation extraction. *Journal of machine learning research*, 3(Feb):1083–1106.

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.

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.