THU_NGN at SemEval-2019 Task 12: Toponym Detection and Disambiguation on Scientific Papers

Tao Qi , Suyu Ge , Chuhan Wu , Yubo Chen , Yongfeng Huang
Tsinghua National Laboratory for Information Science and Technology,
Department of Electronic Engineering, Tsinghua University Beijing 100084, China
{qit16,gesy17,wuch15,chen-yb18,yfhuang}@mails.tsinghua.edu.cn

Abstract

Toponym resolution is an important and challenging task in the neural language processing field, and has wide applications such as emergency response and social media geographical event analysis. Toponym resolution can be roughly divided into two independent steps, i.e., toponym detection and toponym disambiguation. In order to facilitate the study on toponym resolution, the SemEval 2019 task 12 is proposed, which contains three subtasks, i.e., toponym detection, toponym disambiguation and toponym resolution. In this paper, we introduce our system that participated in the SemEval 2019 task 12. For toponym detection, in our approach we use TagLM as the basic model, and explore the use of various features in this task, such as word embeddings extracted from pre-trained language models, POS tags and lexical features extracted from dictionaries. For toponym disambiguation, we propose a heuristics rule-based method using toponym frequency and population. Our systems achieved 83.03% strict macro F1, 74.50 strict micro F1, 85.92 overlap macro F1 and 78.47 overlap micro F1 in toponym detection subtask.

1 Introduction

Toponym resolution is an important task in the natural language processing field and has many applications such as emergency response and social media geographical event analysis(Gritta et al., 2018). Toponym resolution is usually modelled as a two-step task. The first step is toponym detection, which is a typical named entity recognition (NER) task. The second step is toponym disambiguation, which aims to map locations to its coordinates in the real world.

NER is a widely explored task and most NER methods can be applied to toponym detection. For example, Ratinov and Roth (2009) used n-grams, history predictions as the input features of conditional random fields (CRF) for toponym detection. Usually the performance of these methods heavily relies on the quality of hand-crafted features. However, manually selected features may be sub-optimal. Also, these methods cannot effectively exploit contextual information due to the dependency on bag-of-word features. In recent years, many neural network based methods have been proposed for NER. For example, Ma and Hovy (2016) proposed a CNN-LSTM-CRF model for NER. They use CNN layer to learn character features of each word, LSTM layer to learn the contextual word representations and CRF layer to predict the label jointly. Gregoric et al. (2018) proposed Parallel RNN architecture. They split a single LSTM into multiple equally-size ones with a penalty to promote diversity. However, these methods cannot utilize external knowledge to recognize entities, which is usually important to toponym detection. Usually, linguistic knowledge such as part-of-speech and dictionary knowledge may be useful for toponym detection, and they are easy to obtain. Therefore, in this paper, we aim to incorporate these external knowledge sources to enhance our neural model for toponym detection.

Similarly, there are many works on toponym disambiguation. Most of them are rule-based methods. They use some heuristics to rank the candidates and choose the highest one(Gritta et al., 2018). For example, Karimzadeh et al. (2013) used the geographical level(e.g. country, province and city), the Levenshtein Distance and the population of potential candidates to rank the candidate toponym and choose the highest one. However, the result of toponym disambiguation relied on corpus domain and the rule should be reconsidered when applied to different corpus.

For the toponym detection task, we use TagLM(Peters et al., 2017) as the basic model.

Proceedings of the 13th International Workshop on Semantic Evaluation (SemEval-2019), pages 1302–1307
Minneapolis, Minnesota, USA, June 6–7, 2019.

In our model, we first learn word representations from original characters, then learn contextual word representations by a stacked Bi-LSTM network, and finally use a CRF layer to jointly decode the label sequence. To enrich the representations of words, we incorporate various features such as pre-trained word embeddings, POS tags and lexicon features. For the toponym disambiguation task, we design a rule-based heuristics method by using toponym frequency and population to rank candidate toponyms. Our systems achieved 83.03% strict macro F1 in the toponym detection task, 67.21% in the toponym disambiguation task and 61.31% strict macro F1 in toponym resolution.

2 Our Approach

2.1 Toponym Detection

Our model is based on TagLm (Peters et al., 2017). As shown in Fig. 1, our model have three major components, i.e., *character encoder*, *feature concatenation* and *toponym detector*.

Usually, character patterns are important clues for toponym detection. For example, starting with a capital letter (e.g. Eastern Europe), all cased word (e.g. UK) and mixed cased word (e.g. HongKong) are very common in toponym names. Thus, we use a *character encoder* module to learn word representations from original characters. There are two layers in the *character encoder*. The first one is a character embedding layer. It converts each character in a word into a low-dimensional dense vector. The second one is a character-level CNN layer. It was used to capture local contextual information. We also apply a max pooling layer to build word representations by selecting the most salient features.

The *feature concatenation* module is used to concatenate different types of features. There are four types of additional features in our model, i.e., pre-trained word embeddings, pre-trained language model word representations, POS tag representations and lexicon representations. Usually, word embeddings are pre-trained on a large corpus and can provide rich semantic information. Thus, we use pre-trained word embeddings to enrich word representations by incorporating semantic information . However, word embeddings usually do not contain contextual information. Thus, we also incorporate word representations generated by pre-trained language models. Usually, toponyms have specific POS tags such as nouns. Following Wu et al. (2018), we also incorporate POS tag information to guide our model. There are two layers in our model to learn POS tag representations. The first one is a POS tag embedding layer, which learns low-dimensional embedding vectors for POS tags. The second one is a Bi-GRU layer. It was used to learn the syntax structure of sentences and output the hidden POS tag representations. In addition, since many toponyms can be found in toponym databases, lexical features may be useful for toponym detection. Due to our observations, toponym names are more likely to have low occurrence frequency in documents and less number of matched toponyms in the toponym database. Thus we constructed three one-hot vectors as lexical features. First, we counted the number of matched toponyms in the database for every word. They were quantified to different levels and represented by the first one-hot vectors. The second one-hot vectors were used to represent whether the first matched toponym 's names returned from the database were perfect matches. The third one-hot vectors were used to represent quantified occurrence frequency in the training set of every word. Besides, a three-layer feed-forward neural network (FFNN) was used to learn lexical representations for every word as a lexical representation.

The *toponym detector* module aims to predict the label of each word from its representations. There are two submodule in the toponym detector. The first one is a stacked Bi-LSTM network. Usually, global contexts are important for toponym detection. For example, in the sentence "Beijing is the capital of China", the words "capital" and "China" are all informative for toponym detection. Thus, we use a stacked Bi-LSTM network to learn hidden word representations based on global contexts. The second one is a CRF layer, which is used to decode the label sequence jointly (Lafferty et al., 2001). Usually, there is relatedness between the labels of neighbor words. For example, the label "I" (inside) can only appears after "B" (beginning). Thus, we use CRF to do joint label decoding.

2.2 Toponym Disambiguation

Toponym disambiguation is a down-stream task of toponym detection. Due to the lack of dictionary knowledge, it's difficult for a neural network to do toponym disambiguation. Thus, we propose

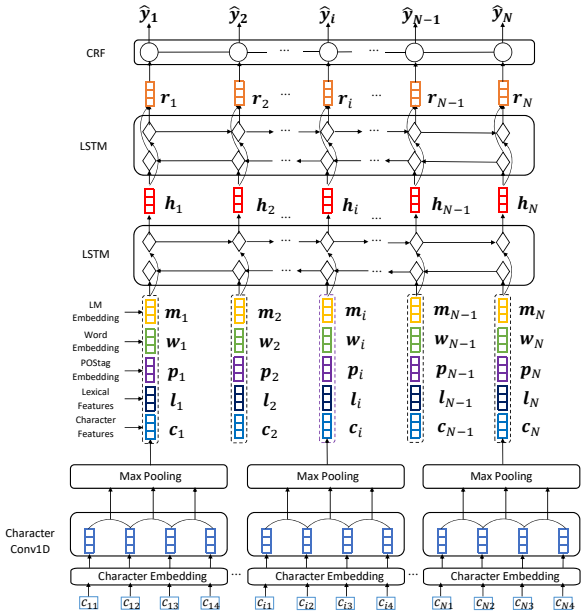


Figure 1: The architecture of our model for toponym detection.

a rule-based heuristic method for toponym disambiguation. An observation is that among the toponym candidates returned by the database, the toponym with higher frequency is more likely to be mentioned. In addition, the toponym with higher population may also have a higher probability to be mentioned. Therefore, we propose a heuristic algorithm named Most Frequency - Most Population (mFmP). If a toponym appears in the train set, we will select the highest frequency id as the output. Otherwise, we will select the toponym with the most population as output.

3 Experiment

3.1 Experimental Settings

We conduct experiments on science reports provided the SemEval-2019 task 12. The data set is composed of 72 full-text journal articles in open access. There are four different metrics to evaluate the prediction performance, i.e., strict macro F1, strict micro F1, overlap macro F1 and overlap micro F1.

In the toponym detection task, we used NLTK[1] for sentence segmentation, word tokenization and POS tagging. We used ELMo(Reimers and Gurevych, 2017) and BERT(Devlin et al., 2018) model to generate 1024-dimensional contextualized word embeddings. We used GeoNames[2] to construct lexical feature. The BIO tagging scheme(Sang and Veenstra, 1999) was used in the toponym detection task. In the toponym disambiguation task, we use GeoNames database to retrieve candidate toponyms.

In our approach, the three word embedding vectors we used (Glove(Pennington et al., 2014), word2vec(Mikolov et al., 2013), fasttext(Bojanowski et al., 2017)) were all 300-dimensional. The dimension of the character embedding was set to 100. The character CNN had 100 filters, and their window size was set to 3. The sizes of the 3-layer FFNN were respectively set to 256, 256, and 128. We set the dimension of POS tag embeddings to 128. The Bi-GRU layer for POS tag representation learning was 64-dimensional. The two Bi-LSTM layers for capturing long-distance and short-distance information were 128-dimensional and 64-dimensional. To mitigate overfitting, we added 20% dropout to each layer. We used Adam as the optimizer for model training.

In our approach, we used transductive learning techniques to further improve the performance of our approach. We first trained our model on the train set, and then applied our model to the test set to generate pseudo labeled data. Finally, we jointly trained our model on the combination of the training and test sets. In addition, we use model ensemble strategy to reduce the uncertainty of our model(Wu et al., 2017). We trained our model for 10 times independently and the final predictions are made by voting.

3.2 Performance Evaluation

In this section, we compare our approach with several baseline methods to evaluate the performance of our approach. The baseline methods are listed as follows. (1) **Baseline**: a baseline system provided by SemEval 2019 task 12(Davy et al., 2019). It uses n-grams as input and a FFNN network to predict label. The input features includes word embedding and character features. (2) **CNN-CRF**: a two-layer CNN and a CRF layer with word embedding for toponym detection. (3) **LSTM-CRF**: a two layer LSTM and a CRF layer with word embedding for toponym detection.

The comparative results are listed in Table 1. According to these experimental results, we have several observations. First, *LSTM-CRF* outperforms *CNN-CRF*. This may be because CNN can

[1]https://www.nltk.org
[2]http://www.geonames.org

method	SMA	SMI	OMA	OMI
baseline	75.56	69.84	80.55	82.60
CNN-CRF	51.28	42.20	61.23	52.51
LSTM-CRF	61.26	47.73	73.26	61.56
Our approach	**84.10**	**82.36**	**91.36**	**90.72**

Table 1: Performance of different toponym detection methods. SMA, SMI, OMA, and OMI respectively denote the strict macro F1, strict micro F1, overlap macro F1 and overlap micro F1.

method	strict macro F1	strict micro F1
baseline	**84.00**	77.59
mFmP	83.58	**78.14**

Table 2: Performance evaluation of toponym disambiguation.

only capture local information, instead, LSTM can utilize global information. This indicates that capturing global contextual information have the potential to improve the performance of toponym detection. Second, the baseline method outperforms *LSTM-CRF* and *CNN-CRF*. This may because *LSTM-CRF* and *CNN-CRF* did not use character-level features, which shows the effect of character-level information on the performance of toponym detection. In addition, the performances of *CNN-CRF* and *LSTM-CRF* are very poor. This may because these two models only use word embedding to enhance the model's semantic information. And this word embedding is not trained on science reports dataset, which may make these two methods lack of semantic information. Third, our approach outperformed all these baseline methods. This is because our approach use pre-trained word embeddings and language model to enhance the semantic information of model, use character-level word representations to capture character patterns of toponym names, and features, i.e., lexicon features and POStag features, to add extract information. This result validates the effectiveness of our model.

The results for toponym disambiguation are listed in Table 2. Baseline method selected toponym with highest population among candidate toponyms. The performance of our method is similar to baseline method. This may be because high frequency toponyms are often with large population.

Method	SMA	SMI	OMA	OMI
WE	77.50	62.61	83.82	69.33
WE+CE	81.75	66.16	85.78	70.16
TagLM	85.04	81.78	90.39	89.61
TagLM+POS	83.64	78.35	90.03	88.87
TagLM+LEX	**85.37**	77.77	90.91	86.57
TagLM+POS+LEX	84.10	**82.36**	**91.36**	**90.72**

Table 3: Influence of different features on the performance of our model. WE, CE, POS, LEX respectively denote toponym detector using word embedding, character encoder, POS tag representations and lexicon representations as input.

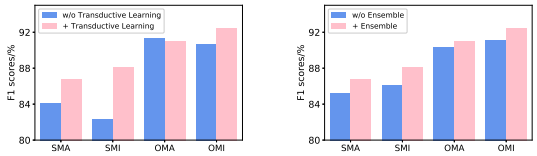
3.3 Influence of Different Features

In this section, we conduct several experiments to evaluate the effect of each type of features we used. We added different types of features to our toponym detector gradually to conduct our experiments. The experimental results are listed in Table. 3. According to Table. 3, we have several observations.

First, the WE+CE method consistently outperforms WE. This indicates that character-level word representations can make our model detect toponym names effectively. Second, the performance of TagLM is much better than the performances of WE and WE+CE. This may because WE and WE+CE method can only obtain semantic information from word embedding and training data, which is not enough. Incorporating word embedding vectors generated by pre-trained language models could enhance the semantic information of our model. Third, after incorporating POS tag embedding or lexical feature into our model separately, the performances of these two models declined. This may be because the POS tag and lexical features of several samples are inaccurate, which incorporate misleading information into our model. Forth, incorporating these two features into our model together improve the performance of our model. This may be because both features have inherent relatedness and our model is more easily to exploit useful information from the combination of both features.

3.4 Influence of Transductive Learning and Model Ensemble

In this section, we conduct several experiments to evaluate the influence of transductive learning and model ensemble on the performance of our model. The experimental results are shown in Figure. 2.



(a) Influence of transductive learning.

(b) Influence of model ensemble

Figure 2: Influence of transductive learning and model ensemble.

According to Figure. 2, we can find both transductive learning and ensemble strategy can improve the performance of our model. This indicates that our model could be more robust by incorporating more training samples and voting scheme.

4 Conclusion

In this paper, we introduce our system participating in the SemEval-2019 task 12. For the toponym detection, we use a TagLM model with various features to enrich word representations. In addition, we use a transductive learning method and ensemble strategy to further improve the performance of our model. For toponym disambiguation, we propose a heuristics rule-based method based on toponym frequency and population. Our systems achieve 83.03% strict macro F1 in toponym detection, 67.21% strict macro F1 in toponym disambiguation and 61.31% strict macro F1 in toponym resolution.

Acknowledgments

This work was supported by the National Key Research and Development Program of China under Grant number 2018YFC1604002, and the National Natural Science Foundation of China under Grant numbers U1836204, U1705261, U1636113, U1536201, and U1536207.

References

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.

Weissenbacher Davy, Magge Arjun, O'Connor Karen, Scotch Matthew, and Gonzalez Graciela. 2019. Semeval-2019 task 12: Toponym resolution in scientific papers. In *Proceedings of The 13th International Workshop on Semantic Evaluation*. Association for Computational Linguistics.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Andrej Zukov Gregoric, Yoram Bachrach, and Sam Coope. 2018. Named entity recognition with parallel recurrent neural networks. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, volume 2, pages 69–74.

Milan Gritta, Mohammad Taher Pilehvar, Nut Limsopatham, and Nigel Collier. 2018. Whats missing in geographical parsing? *Language Resources and Evaluation*, 52(2):603–623.

Morteza Karimzadeh, Wenyi Huang, Siddhartha Banerjee, Jan Oliver Wallgrün, Frank Hardisty, Scott Pezanowski, Prasenjit Mitra, and Alan M MacEachren. 2013. Geotxt: a web api to leverage place references in text. In *Proceedings of the 7th workshop on geographic information retrieval*, pages 72–73. ACM.

John Lafferty, Andrew McCallum, and Fernando CN Pereira. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data.

Xuezhe Ma and Eduard Hovy. 2016. End-to-end sequence labeling via bi-directional lstm-cnns-crf. *arXiv preprint arXiv:1603.01354*.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543.

Matthew E Peters, Waleed Ammar, Chandra Bhagavatula, and Russell Power. 2017. Semi-supervised sequence tagging with bidirectional language models. *arXiv preprint arXiv:1705.00108*.

Lev Ratinov and Dan Roth. 2009. Design challenges and misconceptions in named entity recognition. In *Proceedings of the Thirteenth Conference on Computational Natural Language Learning*, pages 147–155. Association for Computational Linguistics.

Nils Reimers and Iryna Gurevych. 2017. Reporting Score Distributions Makes a Difference: Performance Study of LSTM-networks for Sequence Tagging. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 338–348, Copenhagen, Denmark.

Erik F Sang and Jorn Veenstra. 1999. Representing text chunks. In *Proceedings of the ninth conference on*

European chapter of the Association for Computational Linguistics, pages 173–179. Association for Computational Linguistics.

Chuhan Wu, Fangzhao Wu, Yongfeng Huang, Sixing Wu, and Zhigang Yuan. 2017. Thu_ngn at ijcnlp-2017 task 2: Dimensional sentiment analysis for chinese phrases with deep lstm. *Proceedings of the IJCNLP 2017, Shared Tasks*, pages 47–52.

Chuhan Wu, Fangzhao Wu, Sixing Wu, Zhigang Yuan, and Yongfeng Huang. 2018. A hybrid unsupervised method for aspect term and opinion target extraction. *Knowledge-Based Systems*, 148:66–73.