# DM_NLP at SemEval-2019 Task 12: A Pipeline System for Toponym Resolution

**Xiaobin Wang, Chunping Ma, Huafei Zheng, Chu Liu, Pengjun Xie, Linlin Li, Luo Si**
Alibaba Group, China
{xuanjie.wxb, chunping.mcp, huafei.zhf, chuci.lc,
chengchen.xpj, linyan.lll, luo.si}@alibaba-inc.com

## Abstract

This paper describes DM-NLP's system for toponym resolution task at Semeval 2019. Our system was developed for toponym detection, disambiguation and end-to-end resolution which is a pipeline of the former two. For toponym detection, we utilized the state-of-the-art sequence labeling model, namely, BiLSTM-CRF model as backbone. A lot of strategies are adopted for further improvement, such as pre-training, model ensemble, model averaging and data augment. For toponym disambiguation, we adopted the widely used searching and ranking framework. For ranking, we proposed several effective features for measuring the consistency between the detected toponym and toponyms in GeoNames. Eventually, our system achieved the best performance among all the submitted results in each sub task.

## 1 Introduction

The toponym resolution task is aimed to detect toponyms in scientific papers and link them to entities in a geographical knowledge base (GeoNames[1] in this task). A toponym is a proper name of a place or geographical entity that is named, and can be designated by a geographical coordinate, including cities, countries, lakes or monuments.

We developed an end-to-end toponym resolution system (for subtask 3) which is a pipeline of toponym detection (for subtask 1) and disambiguation (for subtask 2). We model the detection task as a Named Entity Recognition (NER) and address it with popular sequence labeling framework. For disambiguation task, we adopted the searching and ranking framework which is widely used in Entity linking task.

Toponym is a special type of entity similar to the location entity in the general NER task. Thus, the well-studied NER models may be effective for detecting toponyms. The most successful NER models (Chen et al., 2006; Lample et al., 2016; Huang et al., 2015; Yao and Huang, 2016) are sequence labeling models, including the traditional CRF (Conditional Random Field (Lafferty et al., 2001)) and some variants of RNNs (Recurent Neural Networks) proposed recently, like LSTM-CRF, BiLSTM-CRF, BiLSTM-CNN-CRF, etc. In this paper, We utilize the most popular model, i.e., BiLSTM-CRF for toponym detection. Beyond the model, a prevalent pre-training embedding named ELMo is used after fine-tuning. Model averaging and model ensemble are used for avoiding overfitting. Data sets from other NER tasks are exploited to augment the training data. We also proposed a dictionary based method for detecting toponyms in tables separately. Since tables have some peculiarities, i.e., well formatted yet without meaningful context for toponyms in them.

Toponym disambiguation can be seen as a variant of entity linking (EL) problem, which links entity mentions in articles to entities in knowledge base (KB) like Wikipedia. A typical EL system consists of candidate entity generation and ranking as well as unlinkable mention prediction (Shen et al., 2015). The major challenge is that the KB of toponym lacks of background information other than toponym names, types and coordinates. Therefore, we follow the typical EL method (Hoffart et al., 2011) for toponym disambiguation and propose a classification based ranking method. Specifically, We recast the problem as a binary classification task asking that whether a toponym in GeoNames is a link for given toponym. If more than one positives exist, they are ranked according to their confidence scores. Coupled with the classifier, We introduce many features which measure the consistency between toponyms effectively, including name string similarity, candidate

[1] https://geonames.org



*Proceedings of the 13th International Workshop on Semantic Evaluation (SemEval-2019)*, pages 917–923
Minneapolis, Minnesota, USA, June 6–7, 2019. 

attributes, contextual features and mention list features.

Our contributions to this task can be summarized as follows:

- Proposing an approach to process tables separately from the main body.
- Proposing a novel data augment approach to exploit external data.
- Designing many novel and effective features for disambiguation.

## 2 Methodology

### 2.1 Overview

Our system for toponym resolution consists of toponym detection and disambiguation. The former is based on a sequence labeling model and is enhanced with pre-training, model ensemble and data augment. The later is a two-stage approach which obtains candidates by searching and does disambiguation via classification.

### 2.2 Toponym Detection

A scientific article usually contains a main body and tables. Detecting toponyms in these two types of content are different due to toponyms in tables lack of contextual information. Consequently, we adopt two different approaches.

#### 2.2.1 Detection in Main Body

We recast the problem the Toponym Detection in main body as a Named Entity Recognition task and we make use of the BiLSTM-CRF model with the contextual information as input. To alleviate over-fitting, we apply model averaging training strategies. Finally, a voting method is utilized to benefit from multiple models.

**Input Information** Based upon our previous work (Ma et al., 2018) on sequence labeling, our system incorporates four types of linguistic information: Part-of-Speech (POS) tags, NER labels, Chunking labels and ELMo (Peters et al., 2018). The former three are generated by open source tools. In detail, we use Stanford CoreNLP (Manning et al., 2014) to annotate POS tags and NER labels, and use OpenNLP [2] to annotate Chunking labels. These information are represented as distributional vectors which are randomly initialized and trained with the entire model. ELMo

[2] https://opennlp.apache.org/

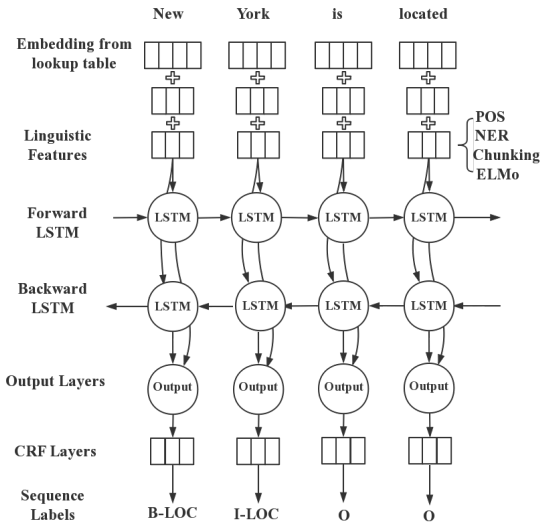


Figure 1: Architecture of BiLSTM-CRF model

is a deep contextualized word representation that models both complex characteristics of word use, and how these uses vary across linguistic contexts. These word vectors are learned functions of the internal states of a deep bidirectional language model (biLM), which is pre-trained on a large corpus of texts. We fine tuned ELMo on the weakly labeled data provided by the organizers, so that the vectors will be adapted to this domain.

**BiLSTM-CRF Model** As illustrated in Figure 1, the entire model consists of five layers: word representation layer, input layer, feature extraction layer, output layer and CRF layer. The word representation layer is a group of BiLSTM with shared parameters. Each BiLSTM corresponds to a word. The BiLSTM takes a sequence of character (characters in a word) embedding as input and concatenates the final hidden states (forward and backward) as the representation of the word. Designing a neural network architecture with character representation as input is appealing for several reasons. Firstly, words which have the same morphological properties (like the prefix or suffix of a word) often share the same grammatical function or meaning. Secondly, a character-level analysis can help to address the out-of-vocabulary problem, Thirdly, capitalization may provide additional information. A recent study (Lample et al., 2016) has shown that BiLSTM is an effective approach to extract morphological information from characters of words, and consequently help to improve the performance

in NER and POS tagging.

The input layer generates the final representation of each word by concatenating three types of vectors, the pre-trained word embedding, the word vector given by the character BiLSTM and the vector of linguistic information (POS label, NE label, chunking label and ELMo vector).

The feature extraction layer is another BiLSTM. RNNs are well-studied solutions for a neural network to process variable length input and have a long term memory. As a variant of RNNs, the long-short term memory (LSTM) unit with three multiplicative gates allows highly non-trivial long-distance dependencies to be easily learned. Therefore, we use a bidirectional LSTM network as proposed in (Graves et al., 2013) to efficiently make use of past features (via forward states) and future features (via backward states) for a specific time frame.

The output layer is a fully connected feed forward network which outputs the probability distribution over all labels.

The CRF Layer is use on the top to decode the appropriate label sequence. For sequence labeling tasks, such as POS tagging or NER, the adjacent labels are often strongly related (e.g. I-ORG cannot follow B-PER or I-LOC in NER tasks like CoNLL2003). CRF model is good at modeling these constraints.

**Model Averaging** Random initialization and shuffling order of training sentences introduce randomization when training a model. During our experiments, we found that model predictions vary considerably even when the same pre-trained data and parameters are used. In order to utilize the power of model ensemble and avoid overfitting problem, we use a script provided by tensor2tensor to average values of variables in a list of checkpoint files generated by BiLSTM-CRF networks.

**Ensemble** By using different pre-trained word embeddings or using different linguistic information, we trained multiple models, we apply an average voting strategy to compute the final decision of our system from all models. Experimental result shows that voting indeed boosts the overall performance.

#### 2.2.2 Detection in Tables

As important components of a scientific article, tables have specific formats:

- They usually begin with the word 'Table'.
- The first line is called the header which indicates the meaning of each column.
- All rows follow the schema defined by the header of the tables.

According to our analysis of the training data, many toponyms are mentioned in tables. Nevertheless, the contexts of these toponyms differ significantly from contexts of toponyms in main body. The later are always meaningful sentences. As a result, performances may drop significantly if a model trained to recognize toponyms in the sentences of the main body is used to recognized toponyms occuring in tables. Thus, we propose a novel approach for detecting toponyms in tables which are processed separately with details as follows:

1. Analyze the mean and variance of words counts (split by space), within a window of text. Decreasing the size of window until the variance is smaller than a threshold.
2. If the word 'Table' is found in the context of the window, take the n-gram within this window as toponym if it exists in GeoNames database.

#### 2.2.3 Postprocess

Rule based postprocessing is applied in the end of the detection step to avoiding errors which occur frequently in development set. The following rules are applied to a toponym for generating possible corrections, which are confirmed and used to replace the original mention by figuring out whether a correction exists in GeoNames.

- If a word of locality, such as eastern, appears before a toponym within three words, we correct the candidate predicted by adding the word of locality to the toponym.
- If a toponym ends with a suffix word (e.g., Province) which indicates an administrative division, we make a candidate correction by removing the suffix when the suffix occurs in a predefined black list.
- If an abbreviation appears after a toponym and the abbriviation consists of of all the capital letters of the words composing the name of the toponym, we include the abbreviation as a new candidate toponym.

### 2.3 Toponym Disambiguation

Our approach for disambiguation has two stages. First, we retrieve possible candidate toponyms from GeoNames database using a search engine with a toponym mention as query. Second, a binary classifier with carefully designed features are applied to each candidate to figure out whether it is the appropriate place that the mention refers to.

#### 2.3.1 Candidate Generation

This stage is based on an offline search engine implemented with Lucene[3]. All GeoNames records were indexed in advance. Then, we search the index with the toponym mentions given by the detection module as queries. In order to ensure higher recall rate, we addressed the alias issue. We expand the query by alternate names and enable fuzzy matching searching.

Alternative names of given toponym mentions are obtained by the following ways:

1. Alternative names recorded in GeoNames dump files, including allCountries, alternatenames, countryInfo.

2. Abbreviations of state names in America given by Wikipedia[4].

3. Alternative names mined by pattern matching from the article where the mention appeared. For example, by using the pattern '<mention>, (<abbr>)' we can get the alternate name 'RSA' of mention 'Republic of South Africa' from sentence 'Republic of South Africa, (RSA)'.

Fuzzy matching is enabled since there are some incorrect spellings in source articles which lead to empty results. However, fuzzy matching introduces noises, so it is enable only if the original query recalls nothing.

#### 2.3.2 Candidate Ranking

We formulate the candidate ranking problem as a binary classification problem. Given a mention detected, several potential candidates are retrieved during candidate generation stage. We take every mention and a candidate pair as input for a binary classifier to decide whether the mention refers to the candidate. We consider the classification confidence as the candidate ranking score $score\langle m, e\rangle$ to select the most likely candidate. To deal with context-poor KB problem, we design information rich features and use the ensemble of model strategy.

**Features** We divide all the features into four groups, i.e., Name String Similarity, Candidate Attributes, Contextual Features and Mention List Features.

1. **Name String Similarity** Following previous work (Shen et al., 2015), we developed features capturing similarity between the candidate's and the mention's name, including Exact Match, Mention Substring of Candidate, Candidate Substring of Mention, Mention Starts Candidate Name, Candidate Starts Mention Name, Jaccard Similarity, Levenshtein Similarity. All names are lowercased in advance and the name of candidate may change into its alternate names if exist.

2. **Candidate Attributes** This set of features are based on target KB's (GeoNames) records and capture some priority of candidate, including Popularity, Number of Ancestors, Code Level.

3. **Contextual Features** Inspired by previous work (Guo et al., 2013), We designed this set of features to measure the contextual similarity between the mention and the candidate. Firstly, for mentions, we take multiple levels of context around mentions in documents as mention-side context, including sentenceparagraph and document level. Secondly, since target KB (GeoNames) lacks context information, we resort to Wikipedia to request candidate's page via API [5]. Considering computation efficiency and avoiding the noise introduced by whole wiki page, we just use the summary (first description paragraph) of the page as candidate-side context, instead of multiple levels. Finally, Bag-of-words representation is employed to mention-side and candidate-side context. Several similarity methods have been explored, including word overlap, cosine similarity and Jaccard similarity.

4. **Mention List Features** We found that the true candidate (or it's ancestor candidates)

[3] https://lucene.apache.org/

[4] https://en.wikipedia.org/wiki/List_of_U.S._state_abbreviations

[5] https://github.com/goldsmith/Wikipedia

may also refer to another mention in the same document. This makes sense because toponyms often co-occur with their child or parent toponyms in medical articles or just occur repeatedly in the same document. We developed so called Mention Neighbors Features, which take all mentions in a document as mention list. Similar to mention-side context, every mention has its sentence, paragraph, and document mention list. We encode the relationship between multi-level mention lists and by checking whether the candidate name, its ancestor name or alternate names occur in the mention lists. This set of features can capture the coherence to some extent.

**Classification Model** We use LightGBM (Ke et al., 2017) as our base model, which gets higher performance compared with other gradient boosting models such as gbdt, xgboost and more traditional models like LR and SVM.

**Ensemble & Stacking** We select different hyper parameters of LightGBM to build a set of base models. Hyper parameters vary in number of estimators, number of leaves, and learning rate. Furthermore, We add a soft-vote classifier as model ensemble, which returns the class label as argmax of the sum of predicted probabilities. Based on all the base models (several LightGBMs, two vote classifiers), we apply a model stacking strategy that takes the outputs (probabilities and labels) of all base models as input and train a simple linear classifier called stacking model and return the stacking model output as the final output.

## 3 Experiments

### 3.1 Toponym Detection

#### 3.1.1 Dataset and Settings

Given 105 medical papers from PubMed Central[6] for developing system, we randomly divided the data into training, development and test set by a ratio of 5:1:1. To avoiding instability of experimental results, we repeat this process 5 times and yield different distributions. All the results shown below are average values among these five distributions.

**Data Augment** The official training data is smaller than the dataset used in general NER task. Therefore, we expanded the training data by selecting external data from CONLL2003 and ontonotes5.0. Sentence containing GPE or LOC entities were selected. A binary classifier [7] was applied to distinguish the external sentences from the official sentences and outputs a confidence score. If the score lower than a threshold, in other words, the external sentence is similar to the official sentence, we add the external sentence into training data. Finally, we obtained 8000 extra training sentences, about 32% of the total training data.

**Preprocessing** Articles are segmented into sentences by NLTK and segmentation errors are corrected based on NER results (generated by CoreNLP). For example, "St. Louis" is split by '.' incorrectly. But it is a location according to NER Results.

#### 3.1.2 Ablation Study

Table 1 shows the ablation study of the detection model. As mentioned above, the baseline model is a Char-LSTM-LSTM-CRF model (Lample et al., 2016). We tried two types of pre-trained embeddings, GloVe (Pennington et al., 2014) and PubMed [8]. Since the PubMed is trained on in-domain data, it achieves better results. Thus, all the rest results are based on embeddings trained on PubMed dataset.

Among the four linguistic information, adding ELMo yields the most improvement, while adding the other three yield a little. we successfully use voting, a simple ensemble method to take advantage of multiple models trained by using different linguistic information, and it works.

All techniques proposed contribute to the performance according to the results. Bring in more training data indeed works but the improvement is feeble. Processing tables separately increases the recall since there are many tables containing toponyms.

The best result is obtained by leveraging all the approaches in combination, it outperforms the baseline model significantly.

### 3.2 Toponym Disambiguation

#### 3.2.1 Dataset and Settings

**Data** The distributions of articles is the same as those experiments of Toponym Detection. Exter-

[6] https://www.ncbi.nlm.nih.gov/pmc/

[7] The training data for this classifier is obtained by mixing the official and external sentences with source information kept. The threshold is chosen by intuition.

[8] trained on PubMed and Wikipedia articles, downloaded from http://bio.nlplab.org/

| Model | | Precision | Recall | F1-score |
|---|---|---|---|---|
| Baseline+PE | GloVe | 85.61 | 82.81 | 84.19 |
| | PubMed | 87.60 | 83.24 | 85.37 |
| Baseline+PE+LF | POS | 87.73 | 83.19 | 85.40 |
| | NER | 87.38 | 83.55 | 85.42 |
| | Chunking | 87.92 | 83.47 | 85.64 |
| | ElMo | 89.40 | 88.34 | 88.87 |
| Baseline+PE+LF+ME | | 89.63 | 88.51 | 89.06 |
| Baseline+PE+DA | | 88.25 | 83.73 | 85.93 |
| Baseline+PE+TP | | 88.36 | 84.78 | 86.53 |
| Baseline+PE+PP | | 87.96 | 83.41 | 85.62 |
| +All | | **90.69** | **89.74** | **90.21** |

Table 1: Experiment results of Detection. Abbreviations: DA, Data Augment; PPE, Pubmed Pre-trained Embedding; PP, Post Process; TP, Table Process; LF, Linguistic Features; ME, Model Ensemble

nal data is not included since they contains no annotation for disambiguation task.

**Hyper-parameters** LightGBM models trained with different hyper-parameters constitute the base model set. The number of estimators varies from 200 to 800, number of leaves from 30 to 50, and the learning rate takes one of 0.05, 0.1. Variance threshold is set as 0.9 at feature selection phase.

#### 3.2.2 Candidate Ranking Results

Table 2 shows the experimental results. We compare the baseline method, single LightGBM model, soft-vote method, and stacking method. The baseline method take the candidate with most population as output.

From Table 2, we can see the LightGBM model beat the baseline method, and model combination strategy improve the performance further. We take outputs of all LightGBM models and soft-vote model as input samples for training a stacking LR model and get the best performance of 89.85%.

For the final run in competition, we chose the stacking method and retrained all base models on the entire train set and predicted on the test set.

#### 3.2.3 Ablation Study

We also conducted an ablation study to investigate the impact of each group of features. From Table 3, we can see Name String Similarity is far below the baseline method(80.45%) and and using the population as a feature is a strong heuristic in fact. Although attribute features take the population as one feature but the classifier using these features still fail to beat the baseline. A reasonable explanation is some other attributes act as noise.

Not surprisingly, Contextual Features play a great role and bring an essential improvement surpassing the baseline. Interestingly, Mention List features, allow a bigger progress over Contextual Features. We think they capture the particularity of toponym disambiguation and some coherence.

| Model | Prec. | Rec. | F1 |
|---|---|---|---|
| baseline | 79.96 | 80.94 | 80.45 |
| lightGBM-single | 89.30 | 87.03 | 88.15 |
| soft-vote | 89.44 | 87.83 | **88.63** |
| stacking | 90.57 | 89.14 | **89.85** |

Table 2: Main results of Candidate Ranking on entire trainset

| | Prec. | Rec. | F1 |
|---|---|---|---|
| +name similarity | 60.40 | 63.06 | 61.70 |
| + attribute | 75.98 | 76.75 | 76.36 |
| + contextual | 86.56 | 85.31 | **85.93** |
| + mention list | 89.30 | 87.03 | **88.15** |

Table 3: Ablation study for Ranking features

## 4 Conclusion and Future works

This paper introduces our system for toponym resolution which is a pipeline of sequence labeling model based detection and classification model based disambiguation. More works are worthy to be done in the future, such as developing a more sophisticated approach for detection toponyms in table, adopting graph-based disambiguation methods and address this task in an end-to-end manner.

## References


Wenliang Chen, Yujie Zhang, and Hitoshi Isahara. 2006. Chinese named entity recognition with conditional random fields. In *Proceedings of the Fifth SIGHAN Workshop on Chinese Language Processing*, pages 118–121.

Alex Graves, Abdel-rahman Mohamed, and Geoffrey Hinton. 2013. Speech recognition with deep recurrent neural networks. In *2013 IEEE international conference on acoustics, speech and signal processing*, pages 6645–6649. IEEE.

Stephen Guo, Ming-Wei Chang, and Emre Kiciman. 2013. To link or not to link? a study on end-to-end tweet entity linking. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1020–1030.

Johannes Hoffart, Mohamed Amir Yosef, Ilaria Bordino, Hagen Fürstenau, Manfred Pinkal, Marc Spaniol, Bilyana Taneva, Stefan Thater, and Gerhard Weikum. 2011. Robust disambiguation of named entities in text. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 782–792. Association for Computational Linguistics.

Zhiheng Huang, Wei Xu, and Kai Yu. 2015. Bidirectional lstm-crf models for sequence tagging. *arXiv preprint arXiv:1508.01991*.

Guolin Ke, Qi Meng, Thomas Finley, Taifeng Wang, Wei Chen, Weidong Ma, Qiwei Ye, and Tie-Yan Liu. 2017. Lightgbm: A highly efficient gradient boosting decision tree. In *Advances in Neural Information Processing Systems*, pages 3146–3154.

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

Guillaume Lample, Miguel Ballesteros, Sandeep Subramanian, Kazuya Kawakami, and Chris Dyer. 2016. Neural architectures for named entity recognition. In *Proceedings of NAACL-HLT*, pages 260–270.

Chunping Ma, Huafei Zheng, Pengjun Xie, Chen Li, Linlin Li, and Luo Si. 2018. Dm_nlp at semeval-2018 task 8: neural sequence labeling with linguistic features. In *Proceedings of The 12th International Workshop on Semantic Evaluation*, pages 707–711.

Christopher Manning, Mihai Surdeanu, John Bauer, Jenny Finkel, Steven Bethard, and David McClosky. 2014. The stanford corenlp natural language processing toolkit. In *Proceedings of 52nd annual meeting of the association for computational linguistics: system demonstrations*, pages 55–60.

Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014. Glove: Global vectors for word representation. In *Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543.

Matthew Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, volume 1, pages 2227–2237.

Wei Shen, Jianyong Wang, and Jiawei Han. 2015. Entity linking with a knowledge base: Issues, techniques, and solutions. *IEEE Transactions on Knowledge and Data Engineering*, 27(2):443–460.

Yushi Yao and Zheng Huang. 2016. Bi-directional lstm recurrent neural network for chinese word segmentation. In *International Conference on Neural Information Processing*, pages 345–353. Springer.