RGCL-WLV at SemEval-2019 Task 12: Toponym Detection

Alistair Plum[1] *, **Tharindu Ranasinghe**[1] *, **Pablo Calleja**[2],
Constantin Orăsan[1], **Ruslan Mitkov**[1]
[1]Research Group in Computational Linguistics, University of Wolverhampton, UK
[2]Ontology Engineering Group, Universidad Politécnica de Madrid, ES
{a.j.plum, t.d.ranasinghehettiarachchige}@wlv.ac.uk
{c.orasan, r.mitkov}@wlv.ac.uk
pcalleja@fi.upm.es

Abstract

This article describes the system submitted by the RGCL-WLV team to the *SemEval 2019 Task 12: Toponym resolution in scientific papers*. The system detects toponyms using a bootstrapped machine learning (ML) approach which classifies names identified using gazetteers extracted from the GeoNames geographical database. The paper evaluates the performance of several ML classifiers, as well as how the gazetteers influence the accuracy of the system. Several runs were submitted. The highest precision achieved for one of the submissions was 89%, albeit it at a relatively low recall of 49%.

1 Introduction

Resolving a toponym, a proper name that refers to a real existing location, is a non-trivial task closely related to named entity recognition (NER) (Piskorski and Yangarber, 2013). For this reason, using an NER system to detect and assign location tags could seem a good way forward. However, NER systems may not be able to detect whether a name refers to a actual location or not (e.g., *London* in *London Bus Company*). In addition, location names are usually ambiguous, which means it is crucial that these are disambiguated in order to assign the correct coordinates.

While in the past the focus in toponym resolution has been on rule and gazetteer driven methods (Speriosu and Baldridge, 2013), more recent approaches also consider ML-based techniques. DeLozier et al. (2015) describe their ML-based approach, which does not require a gazetteer. The approach calculates the geographical profile of each word, which is refined using Wikipedia statistics, and then fed into an ML classifier. Speriosu and Baldridge (2013) also make use of an ML classifier which is text-driven. Geotags of documents are used to automatically generate a training set. Although the two previous approaches used two standard corpora for toponym resolution, consisting of news articles and 19th century civil war texts, there are wide areas of application for toponym resolution. For instance, Ireson and Ciravegna (2010) explore the use in social media, while Lieberman and Samet (2012) attempt to analyse news streams. Spitz et al. (2016) have also used an encyclopaedic dataset, compiled from Wikipedia, WordNet and GeoNames.

The focus of the SemEval 2019 Task 12 was toponym resolution in journal articles from the biomedical domain (Weissenbacher et al., 2019). The articles that had to be processed were case studies on the epidemiology of viruses, meaning that the developed systems can potentially be used to track viruses. The task was composed of three sub-tasks: (1) toponym detection, followed by (2) disambiguation and the assignment of the appropriate coordinates, as well as (3) the development of an end-to-end system.

This paper presents our participation in the first sub-task. Our system performs first a gazetteer look-up for locations, and then uses machine learning (ML) to classify whether or not it represents an actual location. The gazetteers are extracted from the online geographical database GeoNames, whilst the classification is carried out by feeding the context of potential locations in an ML classifier. The rest of the paper presents the system developed (Section 2), followed by its evaluation (Section 3). The paper finishes with conclusions.

2 System Description

The system developed for this task was designed as a pipeline consisting of three stages: text clean-

*The first two authors contributed equally to the paper.

ing, text processing and identification of locations. The rest of this section presents each of these stages. The system has been made available on online. [1]

2.1 Text Cleaning

The first processing stage identifies parts of the text which do not contain any locations that have to be identified according to the task guidelines. These parts include the references section of each text and the information about authors of the journal articles. In addition, the texts also contain genome sequences and abbreviations of chemicals, which resemble abbreviations for locations. Regular expressions were used to replace these text sequences with spaces. We chose to replace the sequences rather than remove them in order to keep the correct offsets of entities which are crucial in the evaluation process.

Not all the genome sequences were correctly identified due to the variability of how they are represented. As a result, not all these sequences were being replaced by spaces. This introduced noise in our processing pipeline. In addition, in some cases the regular expressions for excluding the references section would fail to correctly identify the boundaries of this section. Since this left large amounts of these texts blank, three texts did not have their respective references sections removed.

2.2 Text Processing

Once cleaned, the texts were processed using components from the ANNIE pipeline within GATE (Cunningham et al., 2002, 2011). The ANNIE pipeline was designed for named entity recognition tasks, but for our purpose we used only the tokeniser and gazetteer lookup components.

We produced three different gazetteers. The first one contained all locations from the GeoNames geographical database. The second gazetteer contained a list of cities from GeoNames with a population of over 5,000. The third gazetteer features a list of countries, capitals, and cities with a population larger than 15,000 people extracted from GeoNames. The default list of regions included with ANNIE was also used. A list of US regions as well as their abbreviated forms was added manually.

[1] https://github.com/TharinduDR/SemEval-2019-Task-12-Toponym-Resolution-in-Scientific-Papers

The output of this module was a list of annotations, including tokens boundaries and tokens matching the gazetteer entries. This information is then used by the ML classifier in the next step.

2.3 Identification of Locations

Once the texts are processed, the next task is to detect whether a candidate location really refers to a location. In addition to cases of common nouns which may also be used as a location, there are also cases where the location names were used as adjectives. For example, in the sentence *Other mutations observed in the HA gene of the Kentucky isolates have also been reported by others*, even though the gazetteer identifies *Kentucky* as a location it is actually referring to a virus entity. According to the guidelines, this should not be annotated as a location, making the task quite difficult.

Analysis of examples from the training data indicated that the context of candidate locations can be used to assess whether the detected word is an actual location or not. For this reason, we trained a machine learning model which uses the context of candidates to distinguish between real locations and falsely identified locations by the gazetteer look-up component. For the experiments presented here, we used a window of two words before and two words after the candidate location to obtain its context. More precisely, if the detected word from the gazetteer is ω_i, the context c_i was defined as,

$$c_i = \omega_{i-2} + \omega_{i-1} + \omega_i + \omega_{i+1} + \omega_{i+2} \quad (1)$$

The annotated gold standard provided by the task organisers was used to create a training set which contained both positive and negative instances. Two machine learning approaches were considered for this word window classification task. The first approach was to use traditional machine learning models, while the other approach was to use neural network models.

2.3.1 Traditional ML Approach

There are multiple ways that words can be translated into a numerical representation before they can be used as features for a machine learning model. The commonly used representations convert sequences of words to a bag of words or tf-idf vectors. However, since their introduction, word embedding models (Mikolov et al., 2013) have been widely used as features for text classification tasks and have proven successful. In addition,

they have the capability to represent the context better than tf-idf vectors. For this reason, we used the 300 dimensional word2vec embedding model trained on the Google news corpus.

The word windows had to be represented by a vector that can be fed as features to a machine learning model, while retaining a unique length over all the training and testing examples, in order to be input into a traditional machine learning model. There are many ways to represent a text window with word embeddings. Simply averaging the word embeddings of all words excluding stop words in a text has proven to be a strong baseline or feature across a multitude of tasks, such as short text similarity tasks (Kenter et al., 2016). Following that, the mean of word vectors in a particular word window was calculated in order to represent the whole word window with a vector, which is a 300 dimensional vector in this scenario. The vector calculated was used as features and fed into several machine learning classifiers such as Support Vector Machines (Cortes and Vapnik, 1995), Random Forest classifier (Breiman, 2001) and XGBoost (Chen and Guestrin, 2015). The parameters were tuned using 10-fold cross validation. For the implementation scikit-learn in python 3.6 was used.

2.3.2 Neural Network Architectures

The representation described above performs poorly on classification tasks such as sentiment analysis, because it loses the word order in the same way it happens with the standard bag-of-words model, and fails to recognise many sophisticated linguistic phenomena (Le and Mikolov, 2014). For this reason, the second approach relies on neural networks which receive as input the embedding vectors corresponding to the context, but without performing any modification on it. Keras was used to implement these neural architectures.

Two neural architectures were developed. The first one was adopted from text classification research (Coates and Bollegala, 2018). As depicted in figure 1 it contains variants of Long Short-Term Memories (LSTMs) with self attention followed by average pooling and max pooling layers. It also has a dropout (Srivastava et al., 2014) between 2 dense layers after the concatenate layer. The model was trained with cyclical learning rate (Smith, 2017).

The pooling layers in the first architecture are considered as a very primitive type of routing

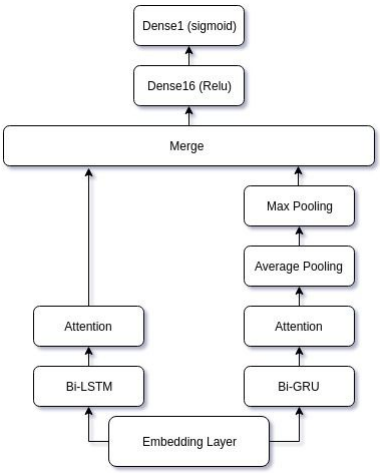

Figure 1: LSTM variant with self attention

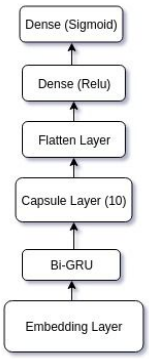

Figure 2: Capsule net architecture

mechanism. The solution that is proposed is a capsule network (Sabour et al., 2017). A capsule network with a bi-directional GRU was also experimented with for this data set. The complete architecture is shown in figure 2. There is a spatial drop out (Tompson et al., 2015) between the embedding layer and bi-directional GRU layer and there is also a dropout (Srivastava et al., 2014) between two dense layers after the capsule layer.

The results and evaluation criteria of both traditional approaches and neural network approaches are reported in the results section 3.

3 Results

3.1 Gazetteers

As described in the previous section, three different gazetteers were tested using the development and training sets. As the machine learning component of the system would make the final prediction, it was important to ensure the maximum number of candidate locations. Therefore, it was

Gazetteer	Precision	Recall	F-Score
GN all	0.2359	**0.7699**	0.3612
GN 5000+	**0.3584**	0.7563	0.4863
GN custom	**0.3546**	**0.7678**	0.4851

Table 1: Gazetteer evaluation results

vital to ensure the highest possible recall, while achieving acceptable precision results.

Table 1 shows the precision, recall and F-score values for each of the gazetteers, described in section 2, run on the training set. Rows one and two had a high recall but low precision, and a higher precision, but lower recall, respectively. Row three shows the results for the final gazetteer. It has the best balance between precision and recall, and was selected for use in the final system.

3.2 Identification of Locations

Locations in the training set were matched using the gazetteers and then extracted together with their respective word window, in order to compile a separate data set. This data was split into a training set and an evaluation set for the machine learning classifiers. The training set consisted of 80% of the total data set and the evaluation set, containing the gold standard annotations from the previous training set, had the rest of the 20%. The accuracy of each machine learning model evaluated on the evaluation set is shown in Table 2. Predictions were considered to be accurate if the machine learning model and the gold standard matched, including correct and incorrect classifications. All other cases were considered to be non-accurate.

Our baseline - a zero-R classifier predicting every instance as a falsely identified location had an accuracy of 71.95%. All of our machine learning models were able to outperform the baseline model significantly, even though the data set is in-balanced. The capsule net architecture, which provided the best performance at an accuracy of 88.73%, was selected for use in the final system.

3.3 Submission Results

After we had determined the best components for the system, the GN custom gazetteer and the bi-GRU + Capsule architecture, the whole system was evaluated on the test set. The submission results are presented in four categories, determined by the organisers. Table 3 shows the results for each. Overall, our system achieves the highest values in the overlap macro class, with the lowest in the strict micro class. The system tends to achieve acceptable precision scores, but at low recall values. This trend can most probably be explained by the fact that many candidate locations are not detected by the gazetteers. Together with the machine learning part discarding some proper locations, this has a dramatic affect on the recall.

ML Model	Accuracy
Zero-R	71.95%
Random Forest	84.21%
SVM	83.44%
XGBoost	85.80%
Bi-LSTM/Bi-GRU + Max Pooling	87.75%
Bi-GRU + Capsule	**88.73%**

Table 2: ML models evaluation results

Test	Precision	Recall	F-Score
Strict macro	0.8280	0.4746	0.6034
Strict micro	0.8168	0.3396	0.4798
Overlap macro	**0.8980**	**0.4969**	**0.6398**
Overlap micro	0.8936	0.3654	0.5187

Table 3: Final submission results

4 Conclusion and Future Work

This paper presented the system we submitted to the *SemEval 2019 Task 12: Toponym resolution in scientific papers*. Evaluation of the system has shown that a pipeline that combines traditional string matching and advanced machine learning can offer promising results. It has demonstrated that a larger size of the gazetteer does not necessarily have a positive effect on performance. It has also made clear that a higher recall value for the gazetteer look-up component could provide a much better basis on which to train machine learning approaches. On the machine learning side, we have demonstrated that employing word embeddings together with state-of-the-art algorithms can be a viable way of classifying toponyms.

Due to time constraints, a large amount of different parameters, as well as optimizing the lookup algorithm and underlying gazetteers were not tested or carried out. For future research we hope to address these problems, so that a better basis on which to train machine learning architectures can be achieved, as well as more deep learning architectures.

References

Leo Breiman. 2001. Random forests. *Machine Learning*, 45:5–32.

Tianqi Chen and Carlos Guestrin. 2015. Xgboost : Reliable large-scale tree boosting system.

Joshua Coates and Danushka Bollegala. 2018. Frustratingly easy meta-embedding - computing meta-embeddings by averaging source word embeddings. In *NAACL-HLT*.

Corinna Cortes and Vladimir Vapnik. 1995. Support-vector networks. *Machine Learning*, 20:273–297.

Hamish Cunningham, Diana Maynard, Kalina Bontcheva, and Valentin Tablan. 2002. GATE: A Framework and Graphical Development Environment for Robust NLP Tools and Applications. In *Proceedings of the 40th Anniversary Meeting of the Association for Computational Linguistics (ACL'02)*.

Hamish Cunningham, Diana Maynard, Kalina Bontcheva, Valentin Tablan, Niraj Aswani, Ian Roberts, Genevieve Gorrell, Adam Funk, Angus Roberts, Danica Damljanovic, Thomas Heitz, Mark A. Greenwood, Horacio Saggion, Johann Petrak, Yaoyong Li, and Wim Peters. 2011. *Text Processing with GATE (Version 6)*.

Grant DeLozier, Jason Baldridge, and Loretta London. 2015. Gazetteer-independent toponym resolution using geographic word profiles. In *Twenty-Ninth AAAI Conference on Artificial Intelligence*.

Neil Ireson and Fabio Ciravegna. 2010. Toponym resolution in social media. In *The Semantic Web – ISWC 2010*, pages 370–385, Berlin, Heidelberg. Springer Berlin Heidelberg.

Tom Kenter, Alexey Borisov, and Maarten de Rijke. 2016. Siamese cbow: Optimizing word embeddings for sentence representations. *CoRR*, abs/1606.04640.

Quoc V. Le and Tomas Mikolov. 2014. Distributed representations of sentences and documents. In *ICML*.

Michael D. Lieberman and Hanan Samet. 2012. Adaptive context features for toponym resolution in streaming news. In *Proceedings of the 35th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '12, pages 731–740, New York, NY, USA. ACM.

Tomas Mikolov, Kai Chen, Gregory S. Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. *CoRR*, abs/1301.3781.

Jakub Piskorski and Roman Yangarber. 2013. Information extraction: Past, present and future. In *Multi-source, multilingual information extraction and summarization*, pages 23–49. Springer.

Sara Sabour, Nicholas Frosst, and Geoffrey E. Hinton. 2017. Dynamic routing between capsules. *CoRR*, abs/1710.09829.

Leslie N. Smith. 2017. Cyclical learning rates for training neural networks. *2017 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pages 464–472.

Michael Speriosu and Jason Baldridge. 2013. Text-driven toponym resolution using indirect supervision. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1, pages 1466–1476.

Andreas Spitz, Johanna Geiß, and Michael Gertz. 2016. So far away and yet so close: Augmenting toponym disambiguation and similarity with text-based networks. In *Proceedings of the Third International ACM SIGMOD Workshop on Managing and Mining Enriched Geo-Spatial Data*, GeoRich '16, pages 2:1–2:6, New York, NY, USA. ACM.

Nitish Srivastava, Geoffrey E. 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.

Jonathan Tompson, Ross Goroshin, Arjun Jain, Yann LeCun, and Christoph Bregler. 2015. Efficient object localization using convolutional networks. *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 648–656.

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