Toponym Detection in the Bio-Medical Domain: A Hybrid Approach with Deep Learning

Alistair Plum, Tharindu Ranasinghe, Constantin Orăsan
Research Group in Computational Linguistics, University of Wolverhampton, UK
{a.j.plum, t.d.ranasinghehettiarachchige, c.orasan}@wlv.ac.uk

Abstract

This paper compares how different machine learning classifiers can be used together with simple string matching and named entity recognition to detect locations in texts. We compare five different state-of-the-art machine learning classifiers in order to predict whether a sentence contains a location or not. Following this classification task, we use a string matching algorithm with a gazetteer to identify the exact index of a toponym within the sentence. We evaluate different approaches in terms of machine learning classifiers, text pre-processing and location extraction on the SemEval-2019 Task 12 dataset, compiled for toponym resolution in the bio-medical domain. Finally, we compare the results with our system that was previously submitted to the SemEval-2019 task evaluation.

1 Introduction

The task of toponym resolution (TR) is a topic in natural language processing (NLP) which is aimed at extracting locations from texts. TR includes the sub-tasks of detecting, disambiguating and finally resolving and assigning coordinates to the proper location in the text. The first step is to detect a location in a text, which can be carried out by more simple means of gazetteer lookup or named entity recognition (NER) (Piskorski and Yangarber, 2013). Next, each detected location that is ambiguous needs to be distinguished and the correct location chosen (Leidner et al., 2004). The final step involves assigning coordinates or applying some other meta-information to clearly distinguish each proposed location (Smith and Crane, 2001; Leidner et al., 2004).

TR can involve many challenges, caused by cases that are not trivial to resolve. This includes, for example, locations contained in phrases such as *London Bus Company*, which contains *London* but refers to an organisation and should therefore not be marked as a location. Furthermore, locations can be contained in other types of phrases, such as genome sequences mentioned in bio-medical texts, as seen in the SemEval-2019 data (Weissenbacher et al., 2019). For example, the location *Henan* in the string *A/Tree sparrow/Henan/4/04* could be omitted from the results, as it is part of the name of a genome. Similar problems can occur where a location might be detected in a character sequence denoting a chemical.

The increasing availability of online resources has been beneficial to TR. On the one hand large-scale geographical databases, such as GeoNames[1], make information about many different locations easily and freely available. On the other hand, readily available mapping services, such as Google Maps[2], allow users to visualise these locations. Although some of these services, such as GeoNames, have been used since the beginnings of TR (Smith and Crane, 2001; Leidner et al., 2004), they are more complete today. A lack of training and evaluation data also existed for some time, mainly reflected by the fact that no standard corpora existed up until a certain period (Leidner, 2004) and that advanced learning techniques could not be used (Smith and Crane, 2001).

Geographic information contained in texts is highly useful and, therefore, the areas where TR is applied are diverse. In the past, TR has been used to catalogue digital libraries (Larson, 1996; Hill et al., 1999) and to make information retrieval techniques spatially aware (Clough,

[1]http://www.geonames.org/
[2]http://www.google.com/maps

https://doi.org/10.26615/978-954-452-056-4_106

2005). A more specialised area included the automatic tracking of biological specimens from different places around the world (Beaman and Conn, 2003). Today, the possible uses range from analysing social media texts (Ireson and Ciravegna, 2010) to news streams (Lieberman and Samet, 2012), where locations of large-scale activity and problematic regions are mapped, respectively. A further example is the bio-medical domain, where the spreading of viruses can be tracked by analysing texts that mention locations (Weissenbacher et al., 2019). However, as the areas of application are varied, the effectiveness of toponym resolvers is also said to vary among different types of text (Gritta et al., 2018).

This paper presents further research into TR, or to be more precise the detection of toponyms. Moreover, this research was carried out in the context of SemEval-2019 Task 12: Toponym Resolution (Weissenbacher et al., 2019) and aimed specifically at subtask 2, which deals only with the detection of toponyms. To the best of our knowledge, prior work on exploring how a machine learning classifier can be used together with relatively simple string matching to detect locations in texts has been limited. Previously, we explored the use of machine learning classifiers to predict a location within short word windows in the context of SemEval-2019 Task 12 (Plum et al., 2019).We employ our system submitted to the SemEval-2019 task as a baseline and make use of the same dataset, consisting of texts from the bio-medical domain Plum et al. (2019). While the system serving as a baseline was reasonably competitive in terms of precision, it did not achieve a high recall. The neural network architectures that are used for this research are novel to this type of task even though some of the architectures have been used for sentence classification tasks like sentiment analysis, spam detection and so on.

The rest of the paper is structured as follows. We present related work in the field first (Section 2), followed by the methodology employed (Section 3). This section includes a description of the dataset, system and network architectures. Section 4 presents the overall results, which are split into classifier results at the sentence level, i.e. disregarding the indexes and therefore not comparable to the SemEval evaluation (Section 4.1) and the overall results at the word level, where explicit indexes are retrieved in order to be evaluated according as in SemEval (Section 4.2). Finally, in Section 5 we come to general conclusions about the project.

2 Related Work

Detecting and resolving locations or toponyms has undergone some changes in its approach. The topic was first dealt with more extensively towards the late 1990s and early 2000s (Larson, 1996; Hill et al., 1999; Smith and Crane, 2001; Leidner et al., 2004). These earliest approaches were aimed mainly at using geographical information for information retrieval, as well as catalogue searches in digital libraries. While the first of these approaches used named entity tagging, as well as specially constructed gazetteers to detect (Larson, 1996; Hill et al., 1999), others went beyond this and used a combination of methods to disambiguate. Smith and Crane (2001) used NE tagging and a gazetteer to detect locations, and disambiguated these using information gathered beforehand. This information, ”local” and ”document” context, is said to include co-occurring words and other locations mentioned throughout the text, respectively. The authors also use ”world knowledge” gathered from other sources, which mainly includes meta information such as coordinates, size, corresponding political entities and so on (Smith and Crane, 2001). Similarly, Leidner et al. (2004) used a combination of simple heuristics, linguistic cues, co-occurrence statistics and discourse information to detect locations and assign coordinates.

TR has shifted from the methods of earlier approaches and followed the trend of using machine learning techniques. Whereas in the past learning techniques lacked data (Smith and Crane, 2001), this is no longer the case. Approaches using machine learning with (indirect) supervision include Hu and Ge (2009) and Speriosu and Baldridge (2013). Hu and Ge (2009) make use of hierarchical structures ensuing from geographical relations, an approach said to perform in an accuracy range of 73.55 to 85.38 percent on an Australian news corpus. Speriosu and Baldridge (2013) on the other hand, present their text-driven approach, which uses context information to resolve toponyms. The classifiers themselves are trained mainly on semi-automatically generated data, obtained primarily from locations tagged in Wikipedia. While the aforementioned approach relies on the availability of gazetteers, a gazetteer-independent approach has been brought forward by DeLozier et al. (2015). This approach, which also relies on a machine learning classifier to disambiguate toponyms, solely uses NER techniques to detect locations, and is said to perform on a state-of-the-art level on the TR-CoNLL (Leidner, 2006) and Civil War corpora (DeLozier et al., 2015).

Most recently, Gritta et al. (2018) have presented a survey of the current state of TR, which they refer to as geoparsing. While the two terms are essentially synonymous, the authors use geoparsing (or geoparsers) to refer to fully fledged end-to-end systems. These include CLAVIN[3], the Edinburgh Parser (Tobin et al., 2010; Grover et al., 2010), Topocluster (DeLozier et al., 2015) and GeoTxt (Karimzadeh et al., 2019). These systems are said to perform at state-of-the-art levels, and are tested on the Local Global Corpus, as well as a corpus compiled by the authors, which is based on Wikipedia and GeoNames data (Gritta et al., 2018). However, it should be mentioned that Topocluster and CLAVIN apply learning techniques. The Edinburgh Parser and GeoTxt rely on NER and heuristics to rank possible candidates.

The evaluation of toponym resolvers is carried out on specifically created datasets or corpora. Leidner (2004) was the first to raise awareness for the need of a gold standard for these purposes. To this end, the paper describes the ongoing effort to create such a corpus, including a custom markup language (TRML) and editor (TAME) (Leidner, 2004). Later, Leidner (2006) describes the resulting corpus of the previous efforts. It is based on news articles, with $6,980$ human-annotated instances of toponyms. The corpus, utilising the CoNLL format, is still used today (DeLozier et al., 2015; Gritta et al., 2018). However, recently concerns have been raised again concerning the availability of datasets for toponym resolution or geoparsing by both Gritta et al. (2018) and Karimzadeh and MacEachren (2019). Gritta et al. (2018) have contributed their own dataset compiled from Wikipedia. In addition, Karimzadeh and MacEachren (2019) present their tool GeoAnnotator which has been developed to aid the compilation of such corpora. The tool is said to not only be useful for the creation of large-scale corpora on a collaborative basis, but also versatile enough to be used for other applications of NLP.

Our System described in Plum et al. (2019) that was submitted to SemEval-2019 Task 12 will serve as the baseline. The approach uses GATE with ANNIE to detect all the occurrences of locations in a text, using custom gazetteers based on GeoNames. Several gazetteers were tested and the best results achieved with a gazetteer of locations with a population of $15,000$ people or more. Following the string matching, two neural network models are used to classify five-word windows around the matched location. For each window a prediction is made whether a real location is contained or not. The method is reported to have a significant drawback, since a five-word context is not enough to carry out proper classification, as the location itself could, for instance, be a multi-word expression. Furthermore, the gazetteer matching carried out beforehand severely limits the overall recall of the approach, as it is not able to detect locations that are written across line-breaks, or are simply not contained in the gazetteer. In contrast to this system, the approach proposed in this paper predicts locations on a sentence-by-sentence basis, then attempts to retrieve the correct index of each location by using a gazetteer lookup.

3 System Description

This section describes the system we developed for detecting toponyms in bio-medical texts. Our approach is based on our system submitted to SemEval-2019 Task 12, described in Plum et al. (2019). It differs mainly in the order of the processing stages, as well as in the architectures that were used. The previous system matches location names using a gazetteer, followed by a machine learning classifier to predict whether the matched location is a proper location or not (Plum et al., 2019). In the present approach, we use a machine learning classifier to predict whether a sentence contains a relevant location first, and on this preselection we perform a gazetteer lookup to identify the specific index range of each location. We also use spaCy NER[4] to compare the effectiveness of the gazetteer.

The approach for the system is split into three steps, which are explained in the following three sections. The first step deals with the preparation of the texts from the dataset. This involves cleaning noisy sections of text and outputting an input

[3]https://clavin.bericotechnologies.com/about-clavin/.

[4]Version 2.1.3, available at https://spacy.io/

file for the machine learning classifier containing all texts split into sentences. Next, each classifier is trained and run in order to obtain predictions at the sentence level. Finally, the output from the classifier is used in conjunction with a gazetteer lookup algorithm (and later spaCy NER) in order to determine the exact indexes of the detected locations.

3.1 Dataset

To ensure comparability with the baseline system, we work with the same dataset from SemEval-2019 Task 12. This dataset is made up of 150 journal articles from PubMed Central and are from the domain of epidemiology (Weissenbacher et al., 2019). As mentioned previously, the main idea behind detecting locations in texts from this domain is to track the spreading of viruses (Weissenbacher et al., 2019). The articles were downloaded as PDFs and converted to plain text using a pdf-to-txt tool by the organisers of the task. The toponyms were manually disambiguated by the organisers and subsequently annotated using the Brat annotator (Stenetorp et al., 2012). Two texts of the training set were removed, as they were unreadable, probably caused by PDF to text conversion problems. Apart from this, we work with the same training and test splits as supplied by the organisers, which are 73 and 45 texts, respectively. It should be pointed out that we had to adjust the annotations of some of the training texts, as these were carried out on texts using CRLF-type line breaks[5] and did not match the indexes read by our system, as it used LF-type line breaks, which lead to the indexes being offset.

The texts had to be prepared for the classification task. As some parts of the texts had been deemed irrelevant by the organisers of SemEval-2019 Task 12 (Weissenbacher et al., 2019), we had to remove these. This includes the references and acknowledgement sections of each article. We also had to remove certain character strings which are specific to texts from the bio-medical domain. Finally, the texts had to be split into sentences and stored with further information in order to be classified in the next step.

[5]CRLF-type line breaks are commonly used in text files created with Microsoft DOS/Windows operating systems. This type of line break uses two characters to denote the end of a line. LF-type line breaks are commonly used on UNIX/Linux-based operating systems, and only use one character to denote the end of a line.

3.1.1 Text Cleaning

The cleaning of the texts is performed in two steps. First, all line breaks are removed and replaced with spaces. This is mainly to deal with sentence splitting and string matching problems that could occur over line breaks. For instance, a line break character between *New* and *York* would lead to this location to be detected as *York*, not as *New York*. The line breaks have to be replaced by one space, as the annotations take line breaks into account and add these to the index range of a location. It should be mentioned that this requires the texts to use LF-type line breaks, as any other type would require a different number of replacement characters. For this dataset, it was ensured that all files conform to this standard.

Next, we carry out more methods to clean the texts. Using the guidelines set out by the task organisers and a brief analysis carried out by Plum et al. (2019), we determined that certain parts of the texts are not relevant for detection. This includes references and certain character and word strings that describe biological genome sequences. As these often include toponyms that were excluded from annotation in the SemEval-2019 task, these could also be disregarded. In order to remove all irrelevant parts but retain indexing consistency, we use regular expressions to find and replace certain text sequences with an equivalent number of spaces. As before, we replace each character with a space in order to ensure that the indexes match up with the annotations.

We wanted to test the effectiveness of our cleaning methods. Therefore, we tested our methods with both types of text: texts where only the line breaks have been removed and texts that have been completely cleaned. During testing it was clear that the performance was better on fully cleaned texts, due to the reduction in seemingly random strings that could be detected (i.e. the string *[..] ACG GGG MA AUA UGC [..]* could produce the match *MA*, as in the U.S. state *Massachusetts*).

3.1.2 Sentence Splitting

As the identification of locations primarily happens at the sentence level, each individual text needs to be split into sentences. We use spaCy in order to complete this task. The sentences are then output to a CSV file, containing information on each sentence's text id, as well as its own specific index range. This information is necessary at the stage of identifying the exact index range of a location and then producing the annotation files for the SemEval evaluation script. For the machine learning training file, a separate column indicating whether or not a location is contained in the sentence is also included.

3.2 Sentence Level Location Identification

Once the texts have been pre-processed, we run a binary classification on the sentences, predicting whether a sentence contains a location, or not. Table 1 shows some examples from the training dataset which is used for this classification task.

Out of the $17,535$ sentences in the training set, only $2,117$ sentences contain locations. This means that the dataset is highly unbalanced, thus making the classification task quite difficult. The increased difficulty was also caused by the language used. As an example, in the row with the ID PMC2857219, shown in Table 1, the sentence contains *Korea* and is not annotated as a location since it is a part of an organisation name. Also, in the row with the ID PMC2857219, *US* is not annotated as a location, as it functions as an adjective to the word soldier. General named entity recognisers such as spaCy or gazetteer matching could possibly falsely detect that these sentences contain a proper location as defined for this task. Furthermore, sentences similar to ID PMC5837706, shown in 1, caused difficulties, as the text in the sentence was not clear. Considering all of these issues, we need an intelligent classifier that detects whether a sentence contains a location or not, by considering the words that appear in the sentence.

We use five different recurrent network architectures to perform the binary classification task on the sentences: pooled Gated Recurrent Unit (GRU) (3.2.1), stacked Long Short-Term Memory (LSTM) with attention (3.2.2), LSTM and GRU with attention (3.2.3), 2D convolution with pooling (3.2.4) and GRU with capsule (3.2.5). Each classifier was run on prepared and cleaned text (as explained in Section 3.1.1). These models were successfully applied to a number of classification tasks such as GRU for sequence labeling Chung et al. (2014), LSTM for semantic similarity and word analogy Coates and Bollegala (2018), and GRU with capsule for toponym detection Plum et al. (2019). Their success in these tasks inspired us to try them for our problem.

3.2.1 Pooled GRU

This model takes pre-trained fasttext embeddings (Mikolov et al., 2018) as a matrix for the input which is comprised of the vertically stacked embedding vectors corresponding to the words present in the sentence. The matrix can be thought of as a sequence of embedded words. Each of these embedding vectors is fed to the bi-directional GRU (Chung et al., 2014) at their respective timestep. The final timestep output is fed into a max pooling layer and an average pooling layer in parallel (Scherer et al., 2010). Following this, the outputs of the two pooling layers are concatenated and connected to a dense layer (Huang et al., 2017) activated with a sigmoid function. Additionally, there is a spatial dropout (Tompson et al., 2015) between the embedding layer and the bi-directional GRU layer to avoid over-fitting.

The network was trained using adam optimiser (Kingma and Ba, 2015), with a reduced learning rate once learning stagnates. This model has been discussed in Kowsari et al. (2019) as a common model to perform text classification tasks.

3.2.2 Stacked LSTM with Attention

As with the previous model, this model takes pre-trained fasttext embeddings (Mikolov et al., 2018) as an input. Each of these embedding vectors are then fed into a bi-directional LSTM (Schuster and Paliwal, 1997). The output of this layer is again fed into a bi-directional LSTM (Schuster and Paliwal, 1997) with self attention (Vaswani et al., 2017). Finally, the output is connected to two dense layers that are (Huang et al., 2017) activated first with a relu function, and then with a sigmoid function.

Again, this network was trained using adam optimiser (Kingma and Ba, 2015), with a reduced learning rate once learning stagnates. We adopted this model from the Toxic Comment Classification Challenge in Kaggle[6].

3.2.3 LSTM and GRU with Attention

This architecture applies a spatial dropout to the embedding layer (Tompson et al., 2015). The output is then fed in parallel to a bi-directional LSTM-layer (Schuster and Paliwal, 1997) with self attention and a bidirectional GRU-layer (Chung et al., 2014) with self attention

[6]https://www.kaggle.com/c/jigsaw-toxic-comment-classification-challenge

id	start	end	sentence	loc?	location
PMC2857219	15686	15753	Dr Jin-Won Song is a professor of microbiology at Korea University.	0	NA
PMC5005937	9817	9928	kConFab recruited multi-generational, multiple-case families through cancer family clinics in Australia.	1	Australia
PMC2857219	14913	14947	These data showed the epidemiologic link between US soldier patients and rodent hosts at the training sites near the Demilitarized Zone in South Korea.	1	South Korea
PMC5837706	14489	14531	(46.9) 395 (39.4) 75 (7.5) 245 (24.4) 329	0	NA

Table 1: Example rows in the training set. Sentences containing a location are represented with 1 in the *loc?* column and 0 otherwise. The *location* column contains the explicit location names contained in the sentence.

(Vaswani et al., 2017). The output from the bi-directional GRU-layer is fed into an average pooling layer and a max pooling layer. The output from these layers and the output of the bi-directional LSTM-layer are concatenated and connected to a dense layer with relu activation. After that, a dropout (Srivastava et al., 2014) is applied to the output and connected to a dense layer activated with a sigmoid function.

While this network was also trained using adam optimiser (Kingma and Ba, 2015), it was trained with a cyclical learning rate (Smith, 2017) this time. Plum et al. (2019) has used this model to predict whether a word window contains a location or not.

3.2.4 2D Convolution with Pooling

The fourth architecture takes a different approach than the previous architectures by using 2D convolution layers (Wu et al., 2018), rather than LSTM- or GRU-layers. The outputs of the embedding layers are connected to four 2D convolution layers (Wu et al., 2018), each with max pooling layers. The outputs of these are concatenated and connected to a dense layer activated with a sigmoid function after applying a dropout (Srivastava et al., 2014).

This network also uses adam optimiser (Kingma and Ba, 2015) and a reduced learning rate once learning stagnates. This model has been used in the Quora Insincere Questions Classification Kaggle competition[7].

[7]https://www.kaggle.com/c/quora-insincere-questions-classification

3.2.5 GRU with Capsule

Most of the previous architectures rely on a pooling layer. However, this architecture uses a capsule layer (Hinton et al., 2018) rather than pooling layers. After applying a spatial dropout (Tompson et al., 2015) the output of the embedding layer is fed into a bi-directional GRU-layer (Chung et al., 2014). The output is then connected to a capsule layer (Hinton et al., 2018). The output of the capsule layer is flattened and connected to a dense layer with relu activation, a dropout (Srivastava et al., 2014) and batch normalisation applied, and re-connected to a dense layer with sigmoid activation.

The capsule network was trained using adam optimiser (Kingma and Ba, 2015), with a reduced learning rate once learning stagnates. This model has been used in Plum et al. (2019) to predict whether a word window contains a location or not.

3.3 Word Level Location Identification

The location predictions made by the ML-classifier at the sentence level are passed as a CSV file to the string matching script. It runs through each sentence, matching locations from a gazetteer. For matching the strings we use a fast and efficient Aho-Corasick algorithm (Aho and Corasick, 1975). The implementation used is available for the Python programming language[8]

We use a large gazetteer that is comprised of the full list of all locations from the GeoNames database[9]. The main idea behind this is that we want to achieve the highest chance of detecting

[8]https://github.com/WojciechMula/pyahocorasick/

[9]http://www.geonames.org/, last accessed 21.05.2019

every location. However, as the gazetteer is so large, is causes a lot of ”noise” during the string matching, including partial matches and numbers that have no meaning. In order to combat this, we apply several filters after finding matches, in order to exclude certain results. We consciously avoided removing anything from the gazetteer itself, as this would be either time-consuming (manual) or could falsely remove desired locations (automatic). Therefore, we use filters to remove all matches with numbers (i.e. *717, 7Palms, 50da*), strings shorter than three characters which are not both uppercase (i.e. *Bl, b1, al* but not *AL, CA, NY*), sub-string matches (i.e. *London* in *Londonderry*) and all lowercase strings (which are usually fragments of location names left in the database, as in *paseo caribe*). The resulting tables are sorted by index, and duplicates removed. Where matches overlap in terms of indexing, we give preference to the longest match. This ensures that in sentences such as *I live in New York*, we detect only *New York* and not *York* (as these are separate entries in the gazetteer).

For comparison purposes, we also employ spaCy to detect locations in the sentences at this stage. We used the standard English web corpus and the spaCy NER algorithm.

4 Results

As our system operates at two levels, we first present the results of location prediction at the sentence level using the five different recurrent network architectures. Next, we present the results of the prediction at the word level. These results are also regarded as our final results, as these predictions yielded the index range of each location, which were evaluated with an evaluation script. The evaluation script that was utilised was the one provided for SemEval-2019 Task 12.

4.1 Sentence Level Prediction

Results of the sentence level predictions for the test set are shown in Table 2. We use precision and recall to evaluate the results. The third model described, LSTM and GRU with Attention, provided the best results for the cleaned text. Despite the dataset being quite unbalanced, the model reported good precision and recall scores of 0.852 and 0.853, which provided a high F1 score, too.

As this is the best model, we use these predictions as the basis for our word level predictions, which are described in the next section. It should be mentioned that we did run some word level predictions on the output of the other classifiers, but as expected the results were always much lower, due to the decreased starting point.

4.2 Toponym Identification

As mentioned previously, we regard the word level predictions as the overall result of the system. The evaluation was carried out in accordance with parameters set out for SemEval-2019 Task 12, featuring strict and overlap categories on macro and micro levels. For the strict measure, predicted locations are only considered as correct if the text span matches the gold standard exactly. For the overlap measure, predictions are considered to be correct if they share a common span of text with the gold annotations. The python script was made available on Bitbucket[10] by the SemEval-2019 Task 12 organizers.

Table 3 shows the results for both the string matching method using gazetteers to extract the locations with a custom script for indexes, as well as the spaCy NER algorithm (only considering locations) and the baseline system submitted to SemEval-2019. Our best results were achieved in the overlap macro classes, and are highlighted in bold. Overall, while we were not able to beat the best precision score of Plum et al. (2019), we came quite close. Nonetheless, we were able to improve the recall significantly, as well as the overall f-score.

The trade-offs that each approach brings with it should become clear when regarding the results.The approach using the GeoNames gazetteer detects a higher number of locations overall. This is due to the simplistic string matching method backed by such a large gazetteer, and comes at the cost of overall precision. The spaCy NER algorithm is much more precise, but is more limited in terms of recall. We find it most likely that this approach does not tag many locations as such, because the texts are still quite noisy, and because we did not train it on our dataset. Due to the small size and unbalanced nature of our dataset, we did not consider training spaCy any further. In the future, given an appropriate dataset from the bio-medical domain, this could perhaps lead to better results.

[10]https://bitbucket.org/dweissen/semevaltask12evaluator/src/master/

Model	Uncleaned			Cleaned		
	P	R	F1	P	R	F1
Pooled GRU	.755	.792	.772	.789	.816	.793
Stacked LSTM with Attention	.795	.784	.789	.826	.796	.813
LSTM and GRU with Attention	.811	.840	.825	**.852**	**.853**	**.852**
2D Convolution with Pooling	.802	.743	.769	.842	.758	.792
GRU with Capsule	.843	.816	.829	.865	.823	.842
All sentences predicted 1	.042	.500	.078	.060	.500	.107
All sentences predicted 0	.457	.500	.478	.439	.500	.468

Table 2: Results for the sentence level classification. We report Precision (P), Recall (R), and F1 for each model (bold indicates the best set of results). Two baseline predictions with all sentences predicted 1 and all sentences predicted 0 are also reported for comparison.

Approach - Level	Strict Detection			Overlap Detection		
	P	R	F1	P	R	F1
Gazetteer - Macro	.524	.791	.631	**.571**	**.840**	**.680**
Gazetteer - Micro	.508	.659	.574	.580	.731	.647
spaCy NER - Macro	.780	.573	.661	**.861**	**.627**	**.726**
spaCy NER - Micro	.726	.413	.526	.823	.468	.597
Baseline - Macro	.828	.474	.603	**.898**	**.496**	**.639**
Baseline - Micro	.816	.339	.479	.893	.365	.518

Table 3: Results for the word level classification. We report the same measures as previously, for the categories described in Section 4. Results are shown for both approaches (bold indicates the best set of results).

5 Conclusion

We have presented a system for toponym detection based on a recurrent network and gazetteer lookup. The approach is novel in that we use the recurrent network to predict whether a sentence contains a location, from which we later extract the exact location by more simpler means. This eliminates the need for a more extensive sequence labelling task, which would require more elaborately annotated training data. Our approach was able to improve on the baseline system presented at SemEval-2019 Task 12 on the same dataset. Albeit not at the same level of precision.

The main conclusion of this paper is that using a hybrid approach, where a machine learning approach is mixed with more traditional gazetteer lookup methods, seems to introduce too many areas where performance is lost. Both in this paper and in Plum et al. (2019) the machine learning or rather neural network architectures on their own perform quite well. However, performing a gazetteer lookup before or after lowers the result significantly. Of course, in both cases these lookups provide necessary information for the system to work, as they provide the index range of each location. The fact that our system performed better in terms of precision when using the popular spaCy NER algorithm, shows that simple gazetteer lookup lacks precision and is probably too simple.

In the future, we would like to introduce an end-to-end system entirely based on machine learning or recurrent network architectures. One emerging approach is to use BERT (Devlin et al., 2018) for token classification which is a fine-tuning model that wraps the BERT model and adds a token-level classifier on top of the BERT model. The recent release of BioBERT (Lee et al., 2019) makes it easier to apply BERT for NER in the bio-medical domain. While a lot more research has been focused on these kind of architectures, we hope to explore tasks other than only sequence labelling. To this end, our aim is to also explore the use of word and context embeddings further.

References

Alfred V. Aho and Margaret J. Corasick. 1975. Efficient string matching: An aid to bibliographic search. *Commun. ACM* 18(6):333–340.

Reed S. Beaman and Barry J. Conn. 2003. Automated geoparsing and georeferencing of Malesian collection locality data. *Telopea* 10(1):43–52.

Junyoung Chung, aglar Gülehre, Kyunghyun Cho, and Yoshua Bengio. 2014. Empirical Evaluation of Gated Recurrent Neural Networks on Sequence Modeling. *CoRR* .

Paul Clough. 2005. Extracting metadata for spatially-aware information retrieval on the internet. In *Proceedings of GIR'05*.

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

Grant DeLozier, Jason Baldridge, and Loretta London. 2015. Gazetteer-independent toponym resolution using geographic word profiles. In *Proceedings of AAAI 2015*.

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

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

Claire Grover, Richard Tobin, Kate Byrne, Matthew Woollard, James Reid, Stuart Dunn, and Julian Ball. 2010. Use of the Edinburgh geoparser for georeferencing digitized historical collections. *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences* 368(1925):3875–3889.

Linda L Hill, James Frew, and Qi Zheng. 1999. Geographic names. *D-lib Magazine* 5(1):17.

Geoffrey E. Hinton, Sara Sabour, and Nicholas Frosst. 2018. Matrix capsules with EM routing. In *Proceedings of ICLR 2018*.

You-Heng Hu and Linlin Ge. 2009. *A Supervised Machine Learning Approach to Toponym Disambiguation*, Springer London, pages 117–128.

Gao Huang, Zhuang Liu, and Kilian Q. Weinberger. 2017. Densely Connected Convolutional Networks. *Proceedings of IEEE CVPR 2017* .

Neil Ireson and Fabio Ciravegna. 2010. Toponym Resolution in Social Media. In *Proceedings of ISWC 2010*.

Morteza Karimzadeh and Alan M. MacEachren. 2019. GeoAnnotator: A Collaborative Semi-Automatic Platform for Constructing Geo-Annotated Text Corpora. *ISPRS International Journal of Geo-Information* 8(4).

Morteza Karimzadeh, Scott Pezanowski, Alan M. MacEachren, and Jan O. Wallgrn. 2019. Geotxt: A scalable geoparsing system for unstructured text geolocation. *Transactions in GIS* 23(1):118–136.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A Method for Stochastic Optimization. *Proceedings of CoRR 2015* .

Kamran Kowsari, Kiana Jafari Meimandi, Mojtaba Heidarysafa, Sanjana Mendu, Laura E. Barnes, and Donald E. Brown. 2019. Text Classification Algorithms: A Survey. *Information* 10(4).

Ray R Larson. 1996. Geographic information retrieval and spatial browsing. *Geographic information systems and libraries: patrons, maps, and spatial information* .

Jinhyuk Lee, Wonjin Yoon, Sungdong Kim, Donghyeon Kim, Sunkyu Kim, Chan Ho So, and Jaewoo Kang. 2019. Biobert: a pre-trained biomedical language representation model for biomedical text mining. *arXiv preprint arXiv:1901.08746* .

Jochen L Leidner. 2004. Towards a reference corpus for automatic toponym resolution evaluation. In *Proceedings of GIR'04*.

Jochen L. Leidner. 2006. An evaluation dataset for the toponym resolution task. *Computers, Environment and Urban Systems* 30(4):400 – 417. Geographic Information Retrieval (GIR).

Jochen L Leidner et al. 2004. Toponym resolution in text:Which Sheffield is it?. In *Proceedings of GIR'04*.

Michael D. Lieberman and Hanan Samet. 2012. Adaptive Context Features for Toponym Resolution in Streaming News. In *Proceedings of ACM SIGIR*.

Tomas Mikolov, Edouard Grave, Piotr Bojanowski, Christian Puhrsch, and Armand Joulin. 2018. Advances in Pre-Training Distributed Word Representations. In *Proceedings of LREC 2018*.

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

Alistair Plum, Tharindu Ranasinghe, Pablo Calleja, Constantin Orasan, and Ruslan Mitkov. 2019. RGCL-WLV at SemEval-2019 Task 12: Toponym Detection. In *Proceedings of SemEval-2019*.

Dominik Scherer, Andreas C. Müller, and Sven Behnke. 2010. Evaluation of Pooling Operations in Convolutional Architectures for Object Recognition. In *Proceedings of ICANN 2010*.

Mike Schuster and Kuldip K. Paliwal. 1997. Bidirectional recurrent neural networks. *IEEE Trans. Signal Processing* 45:2673–2681.

David A. Smith and Gregory Crane. 2001. Disambiguating Geographic Names in a Historical Digital Library. In *Proceedings of ECDL 01*.

Leslie N. Smith. 2017. Cyclical Learning Rates for Training Neural Networks. In *Proceedings of IEEE WACV 2017*.

Michael Speriosu and Jason Baldridge. 2013. Text-driven toponym resolution using indirect supervision. In *Proceedings of ACL 2013*.

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

Pontus Stenetorp, Sampo Pyysalo, Goran Topić, Tomoko Ohta, Sophia Ananiadou, and Jun'ichi Tsujii. 2012. BRAT: a web-based tool for NLP-assisted text annotation. In *Proceedings of EACL 2012*.

Richard Tobin, Claire Grover, Kate Byrne, James Reid, and Jo Walsh. 2010. Evaluation of georeferencing. In *Proceedings of GIR'10*.

Jonathan Tompson, Ross Goroshin, Arjun Jain, Yann LeCun, and Christoph Bregler. 2015. Efficient object localization using Convolutional Networks. *Proceedings of IEEE CVPR 2015* .

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention Is All You Need. In *Proceedings of NIPS*.

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 SemEval-2019*.

Yunan Wu, Feng Yang, Ying Liu, Xuefan Zha, and Shaofeng Yuan. 2018. A Comparison of 1-D and 2-D Deep Convolutional Neural Networks in ECG Classification. In *Proceedings of IEEE Engineering in Medicine and Biology Society*.