RESEARCH ARTICLE

Toponym Resolution with Deep Neural Networks

Ricardo Custódio

Toponym resolution, i.e. inferring the geographic coordinates of a given string that represents a placename, is a fundamental problem in the context of several applications related to geographical information retrieval and to the geographical information sciences. The current state-of-the-art relies on heuristic rules, together with machine learning methods leveraging simple linear methods. This article advances an approach leveraging Gated Recurrent Units (GRUs), a type of recurrent neural network architecture that can be used for modeling sequential data, to build representations from the sequences of words and characters that correspond to the strings that are to be associated with the coordinates, together with their usage context (i.e., the surrounding words). These representations are then combined and passed to feed-forward nodes, finally leading to a prediction decision. The entire model can be trained end-to-end with a set of labeled placename occurrences, e.g., collected from Wikipedia. We present the results of a wide-ranging evaluation of the performance of the proposed method, using previous works datasets. The model achieved some interesting results, sometimes even outperforming state-of-the-art toponym resolution methods.

Keywords: toponym resolution; deep neural networks; recurrent neural networks; geographic information retrieval

1. Introduction

Recognizing location names (i.e., toponyms) mentioned in textual documents, and associating those names with their respective coordinates of latitude and longitude, is an important problem in the context of automatically processing textual documents, with various applications related to the access, retrieval and exploration of geographical information embedded in text (i.e., in the context of systems for the visualization of information, expressed in text documents, under the form of maps), or as a support for studies in the humanities and/or social sciences (Grover *et al.* 2010, Gregory *et al.* 2015).

The names of locations are usually ambiguous expressions, since these names can correspond to distinct geographical referents (e.g., the name *Lisboa* is associated to dozens of specific geographic locations, besides the capital city of Portugal), and since the same location can often be associated to various alternative names (e.g., the names *Porto* and *Invicta* can both be used as references to the city of Porto in the north region of Portugal).

Most previous work coming from the areas of text processing and information retrieval,

focused on the problem of toponym resolution, leveraged heuristic rules (Leidner 2008, Grover *et al.* 2010). Although significant progress has been made in the general task of named entity resolution (Ji and Grishman 2011, Shen *et al.* 2015), there are relatively few studies focusing on the utilization of statistic learning techniques as a way to approach the specific problem of toponym resolution.

In this article, we present a novel toponym resolution approach to predict geographic coordinates of toponyms. Our solution leverages a deep neural network architecture, with parameters learned from training data (Schmidhuber 2015, LeCun *et al.* 2015, Goodfellow *et al.* 2016). The network uses Gated Recurrent Units (GRUs), a type of recurrent neural network architecture for modeling sequential data, that was originally proposed by Chung *et al.* (2014) to build representations from the sequences of words and characters that correspond to the strings that are to be associated with the coordinates. These representations are then combined and passed to a sequence of feed-forward nodes, finally leading to a prediction. The entire model can be trained end-to-end through the back-propagation algorithm in conjunction with the Adam optimization method (Rumelhart *et al.* 1988, Kingma and Ba 2015), provided access to a training set of labeled sentences.

Besides advancing this novel method, we also present the results of a wide-ranging comparative evaluation on the performance of toponym resolution methods, leveraging previous works datasets (Leidner 2008, DeLozier *et al.* 2016, Lieberman *et al.* 2010a). This paper resulted in the creation of a dataset composed by 600 thousand sentences extracted from the English Wikipedia. Results show that, in some cases, the proposed method can outperform toponym resolution methods used in previous studies (Ardanuy and Sporleder 2017, DeLozier *et al.* 2015).

The remainder of this article is organized as follows: Section 2 presents previous research in the area. Section 3 describes the proposed deep learning method, detailing the neural network architecture that was used for resolving toponyms, and discussing the model training procedure. Section 4 presents the experimental evaluation, detailing the general protocol, datasets and evaluation metrics, and also presenting and discussing the obtained results. Finally, Section 5 concludes the article with a summary of the most interesting findings and with possible paths for future work.

2. Related Work

This section overviews important related work in the area of toponym resolution. Previous publications are divided into heuristic methods, machine learning methods, and grid-based methods.

2.1. *Heuristic Methods*

Most previous work in the area follows the idea that, given a document with identified toponyms, a series of pre-defined rules can be used over each toponym to determine which location it corresponds to. These rules usually rely on gazetteers, which are dictionaries for locations (e.g., Geonames or the Getty Thesaurus) that contain several metadata fields in each entry (e.g., population, area, administration level, etc.). Occasionally, gazetteers are specifically created by the authors for the context of their toponym resolution work. For example, Lieberman *et al.* (2010b) used a *prominence* heuristic to disambiguate comma groups, i.e., lists of toponyms found in documents, separated by some sort of token (e.g. "and", "or", ",", etc.). The *prominence* heuristic consists in checking if all toponyms in a

group have a prominent location interpretation (i.e. population $\geq$ 100k inhabitants) and, if so, resolve the toponyms in the group accordingly.

Leidner (2007) surveys several heuristics used in toponym resolution systems:

- *Largest population.* Given a set of candidates for a toponym, the method simply chooses the candidate with the largest population).
- *Superordinate mention.* Disambiguates toponyms by finding a common higher administrative level referent (i.e., state, continent, country) for the placenames in the same document.
- *Spatial minimality.* Assumes that, the smallest region that is able to ground a whole set of toponyms mentioned in some span of text is the one that gives them their interpretation.

It is common to observe the same heuristics being used in different works. For example, Lieberman *et al.* (2010b) and Lieberman *et al.* (2010a) also use a similar version of *spatial minimality* heuristic from Leidner (2007), which the authors named *proximity* heuristic.

2.2. *Machine Learning Methods*

Another approach to resolve toponyms includes using heuristics, such as those from the previous section, as features in machine learning algorithms.

Lieberman and Samet (2012) addressed the use of adaptive context features in geotagging algorithms for streaming news. Those adaptive context features are based on computing features within a window of context around each toponym. Their toponym resolution method assigns a decision on whether for a given toponym/interpretation pair (t, l_t), the disambiguation candidates l_t are drawn from a gazetteer, the pair is true or not. Random forests (Breiman 2001) were used for classifying each pair, according to descriptive features. This method constructs many decision trees based on different random subsets of the dataset, sampled with replacement. Each decision tree is constructed using random subsets of features from the training feature vectors. To classify a new feature vector, each tree in the forest votes for the vector's class, and the consensus is taken as the result. The authors considered several baseline toponym resolution features in their methods, namely the ones used in Lieberman *et al.* (2010a).

Santos *et al.* (2014) addressed the problem of toponym resolution by first identifying toponyms using Standford's NER system, and then ranking the candidates through a procedure inspired on previous work in entity linking (Anastácio *et al.* 2011), which involves the following steps:

- **Query expansion**: Given a reference, expansion techniques are applied to try to identify other names in the source document that reference the same entity. For example, *NY* is a reference for *New York* and *US* for *United Sates.*
- **Candidate generation**: This step looks for similar entries to the query in a gazetteer built from Wikipedia, and returns the top 50 most likely entries, according to an n-gram retrieval model supported by a Licene[1] index. The gazetteer is based on an English Wikipedia[2] subset containing all the geotagged pages, plus all pages categorized in DBPedia[3] as corresponding to either persons, organizations and locations.

[1] http://lucene.apache.org/index.html
[2] https://dumps.wikimedia.org/index.html
[3] http://dbpedia.org/index.html

- **Candidate ranking**: The LambdaMART learning to rank algorithm (Burges 2010), as implemented in the RankLib[4] library, is used to sort the retrieved candidates according to the likelihood of begin the correct referent. The ranking model leverages on a total of 58 different ranking features for representing each candidate. These features vary from authority features (e.g., the PageRank score of the candidate, computed over Wikipedia's link graph) to textual similarity (e.g., cosine similarity between tf-idf representations for the query document and for the candidate's textual description in Wikipedia), including also geographical features (e.g., spatial minimality heuristics).
- **Candidate validation**: It may be the case that the correct referent is not given in the gazetteer. This step decides if the top ranked referent is an error, trough a random forest classifier that reuses the features from the previous step, and considers additional features for representing the top ranked referent, such as the candidate ranking score, or the results from well known outlier detection tests, that try to see if the top ranked candidate is significantly different from the others.

The geographical features are particularly important given the purpose of toponym resolution, and the ones used by the authors were very similar to the heuristics used in previous work (Leidner 2008, Lieberman *et al.* 2010a,b).

Similarly to Santos *et al.* (2014), Ardanuy and Sporleder (2017) proposed a weakly-supervised method named **GeoSem** that combines the strengths of previous work on toponym resolution and entity linking, by exploiting both geographic and semantic features. The authors start by building a knowledge base of locations, composed by georeferenced articles from Wikipedia complemented with information from Geonames. Each location has a list of alternative names, selected geographic features (e.g., latitude, longitude, population and the country where the location is situated) and semantic features (e.g. context of words extracted from parts of the body of the Wikipedia article). Leveraging the knowledge base, the authors attempt to select the right candidate location with the given toponym. Their method distinguishes between local features (i.e., features that measure the compatibility of a candidate location with the referent of a toponym in a text without regard to compatibility with co-occurring toponyms) and global features (i.e., features that take into account the interdependence between entities to measure the compatibility of a candidate), taking inspiration on previous work by Han *et al.* (2011). The local and global features are then combined and fed into the method, that decides which is the most likely candidate for a given toponym.

2.3. *Grid-based methods*

Grid-based toponym resolution methods, are based on dividing the Earth's surface in a grid of cells, afterwards classifying toponyms according to which cell they belong to. Speriosu and Baldridge (2013) proposed 3 different grid-based toponym resolvers, namely **TRIPDL**, **WISTR** and **TRAWL**.

TRIPDL is an acronym for Toponym Resolution Informed by Predicted Document Locations. This method divides the Earth's surface according to a 1^{o} by 1^{o} grid of cells, and then learns language models for each cell from Wikipedia geolocated articles (i.e., from a collection of Wikipedia pages referred to as the GeoWiki dataset). It then computes the similarity between a document d, represented as a distribution over words, against each cell, and chooses the closest one, using the Kullback-Lieber (KL) divergence

[4]http://people.cs.umass.edu/ vdang/ranklib.html

(Kullback and Leibler 1951). Then, the authors normalize the values of the KL-divergence to obtain a probability $\mathrm{P}(c|d)$ that is used for all toponyms t in d, to define the following distribution:

$$\mathrm{P}_{DL}(l|t,d) = \frac{\mathrm{P}(c_l|d)}{\sum_{t' \in G(t)} \mathrm{P}(c_{t'}|d)} \tag{1}$$

In the previous equation, $G(t)$ is the set of the locations l for toponym t in the Geonames gazetteer, and c_l is the cell that contains l. To disambiguate a toponym, TRIPDL chooses the location that maximizes $\mathrm{P}(c|d)$.

WISTR is an acronym for Wikipedia Indirectly Supervised Toponym Resolver, and this method extracts training instances automatically from the Geowiki dataset to learn text classifiers. It begins by detecting toponyms in Geowiki using the OpenNLP NER[1] system. Then, for each toponym, it gets all candidate locations from the geonames gazetteer. The candidate location that is closest to the Wikipedia article's location is used as the label for the training instance. Context windows of twenty words w to each side of each toponym are used as features. After all the relevant instances are extracted, they are used to train logistic regression classifiers $\mathrm{P}(l|t,w)$ for location l and toponym t. The location that maximizes this probability is chosen to disambiguate a new toponym.

TRAWL is an acronym for Toponym Resolution via Administrative levels and Wikipedia Locations and this last approach corresponds to a hybrid of TRIPDL and WISTR that gives preference to locations more administratively prominent. This means that, for example, if a country and a city have the same name, the method will give preference to the country. TRAWL selects the optimal candidate location $\hat{l}$ according to:

$$\hat{l} = \arg\max_{l} P(a_l|t)(\lambda_t P(l|t,c_t) + (1-\lambda_t) P_{DL}(l|t,d) \tag{2}$$

In the previous equation, $\mathrm{P}(a_l|t)$ is the administrative level component and is given by the fraction of the representative points of location $\hat{l}$ and representative points for all locations $l \in t$. All cities have only one representative point in geonames, so this will give higher probability to states and countries because they contain usually thousands of points. The rest of the Equation 2 is a linear combination of WISTR and TRIPDL. Since the authors are more confident that WISTR will give the right prediction, they assign a weight λ_t to the local context distribution that is given by:

$$\lambda_t = \frac{f(t)}{f(t)+C} \tag{3}$$

In the previous equation, $f(t)$ is the fraction of training instances for toponym t of all instances extracted from Geowiki, while the constant C is set experimentally.

To evaluate the results, the authors used yet another resolver (i.e., a method named Spatial Prominence via Iterative Distance Evaluation and Reweighting, with the acronym **SPIDER**) as a baseline. SPIDER assumes that toponyms in the same document tend to refer to nearby locations, and gives preference to more prominent locations (i.e. locations

[1] http://opennlp.apache.org

that tend to get selected more often in a corpus). The authors also used two simpler baseline resolvers: RANDOM (i.e., randomly select a location in the candidate locations) and POPULATION (i.e., select the candidate location with the highest population).

To evaluate the different resolvers, the authors computed the mean and median of the distance between the correct and predicted locations for each toponym. Precision and recall were also used when dealing with NER-identified toponyms. The authors used the TR-CoNLL (i.e., articles about international events, used primarily in the CoNLL-02 competition on named entity recognition) and CWar (i.e., a collection of documents about the American Civil War[1]) corpora to support their experiments.

The authors concluded that their resolvers outperformed standard minimality resolvers. Overall, the resolver with the best results, both in accuracy, precision and recall, was the WISTR resolver, although in the CWar corpus this method had to be combined with SPIDER to achieve a better accuracy.

DeLozier *et al.* (2015) developed another language modeling method which, they named **TopoCluster**. Their approach identifies geographic clusters for every word, learned from Wikipedia articles extracted from the Geowiki dataset, and then selects the strongest overlapping point with the clusters for all words in a toponym's context.

Like in other previous works (Speriosu and Baldridge 2013), the authors start by dividing the Earth's surface in a grid of cells. Their grid cells are spaced with .5º and they cover an area that spans from latitude 70ºN to 70ºS. The method also ignores every cell that is not within .25º from a land mass.

The authors start by computing the Local Getis-Ord G_{i^*} statistic (Ord and Getis 1995) to measure the strength of association between words and geographic space. This method creates a geographically aggregated and smoothed likelihood of seeing each word at certain points in geographic space. The G_{i^*} statistic is given by:

$$G_i^*(x) = \frac{\sum_{j=1}^{n} w_{ij} x_j}{\sum_{j=1}^{n} x_j} \tag{4}$$

In the previous equation, w_{ij} is a kernel defining the association between a cell i and a document location j. This has the effect of smoothing the contributions from each document according to their proximity to i. The parameter x_j is a measure of strength between a word x a document location j.

The output of these calculations is a matrix of statistics where columns are grid cells and rows are vectors $g^*(x)$ of each word in the vocabulary.

To disambiguate a toponym z, context windows c were extracted from the different corpora, composed by 15 words left and right from z. Those context windows, contain toponyms t and non-toponym words x that were separated from each other. Then, the following weighted sum is computed:

$$g^*(z, c) = \theta_1 \vec{g}^*(z) + \theta_2 \sum_{t \in c} \vec{g}^*(t) + \theta_3 \sum_{x \in c} \vec{g}^*(x) \tag{5}$$

[1] http://www.perseus.tufts.edu/hopper/

In the previous equation, the parameters θ_1, θ_2 and θ_3 weight the contribution of the different types of words (z, t, x). The chosen location is the cell i that maximizes the value of $g * (z, c)$. These weights are determined by trial and error and the combination that gets the best accuracy results is chosen. The chosen location is the grid cell with the largest value in $g^*(z, c)$, which represents the most strongly overlapped point in the grid given all words in the context.

A version of TopoCluster using a gazetteer named **TopoClusterGaz** was also created. This method forces place names to match entries in an hybrid gazetteer built from Geonames and from Natural Earth[1] data.

A domain adaptation (i.e., suit a model learned from a source data distribution to a different target data distribution) of the G_{i^*} statistic can be done for the different corpora. This corresponds to the following equation:

$$\vec{g}^* = \lambda \vec{g}^*_{InDomain} + (1 - \lambda)\vec{g}^*_{GeoWiki} \tag{6}$$

To determine the λ values for the different corpora, the authors varied the value between 0 and 1 and verified if there is an improvement in the system's accuracy. The λ value is then set accordingly.

The corpora used in the experiments were the TR-CoNLL, CWar and LGL datasets. The authors concluded that the base TopoCluster$_{\lambda=0}$ performed poorly on every corpus, but there was a substantial improve on its results when combined with in-domain data. Overall, TopoClusterGaz with domain adaptation got the best results in every corpus that were considered for the tests (i.e., LGL, CWar and TR-CoNLL).

3. The Proposed Approach

In contrast to most previous work on toponym resolution, which leveraged heuristic rules as features within machine learning methods, in this article we propose a novel approach, which leverages the supervised training of a deep neural network that can directly predict the geographic coordinates of a given toponym. A large set of sentences, each one containing a toponym associated with its geographic coordinates is used to infer the parameters of the neural network. This network takes a sentence (represented by embeddings of words and another representation as a sequence of characters) and a toponym (represented by a sequence of characters) belonging to that sentence as input, and outputs the geographic coordinates. After training, the model can be used to predict the coordinates of a given toponym, occurring in a previously unseen document. This section describes the specific architecture that was proposed for toponym resolution.

3.1. *Toponym Resolution Using a Deep Neural Network*

The specific neural network architecture that is proposed in this article for addressing the toponym resolution problem, where recurrent nodes are perhaps the most important components, is illustrated in Figure 1. This architecture takes its inspiration on models that have been previously proposed for natural language inference and for computing sentence similarities (Bowman *et al.* 2015, Rocktäschel *et al.* 2016, Yin and Schütze 2015,

[1] http://www.naturalearthdata.com/

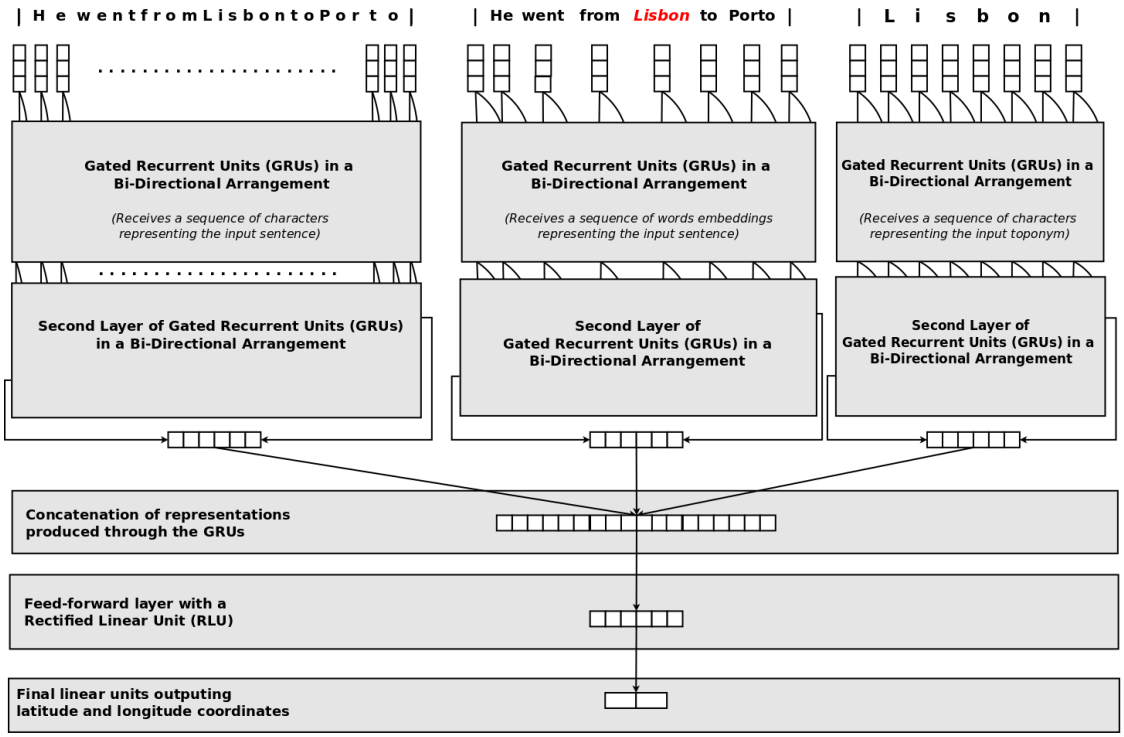

Figure 1.: The neural network architecture proposed to address toponym resolution.

Wan *et al.* 2016, Liu *et al.* 2016, Mueller and Thyagarajan 2016), as well as for computing toponym similarity in the context of duplicate detection (Santos *et al.* 2017)

The input to the network are three sequences. One represents the toponym as a sequence of embeddings of characters; the context in which the toponym appears (i.e., the words surrounding the toponym) are represented with embeddings of words, and finally a representation of the context as embeddings of characters. For the word embedding representation we leveraged 300-dimensional word embeddings, namely, the word2vec pre-trained Google News corpus[1].. The embeddings layer is initialized with basis on these pre-trained values, and then adjusted during model training.

The input representations are provided to bi-directional GRUs, which produce a vector of embeddings, for each sequence. Bi-directional GRUs work by concatenating the outputs of two GRUs, one processing the sequence from left to right and the other from right to left (Schuster and Paliwal 1997). Our neural network architecture actually uses two different layers of bi-directional recurrent units. The first bi-directional GRU layer generates a sequence of real-valued vectors, that is then passed as input to the second bi-directional GRU layer. The second bi-directional GRU layer outputs a single embedding for the input, resulting from the concatenation of the last outputs that are produced by the GRUs that process the input sequences in each direction

The three embeddings produced by the bi-directional GRU layers are then concatenated and passed as input to a first layer that uses a simple combination of the inputs together with a non-linear activation function (i.e., a rectified linear unit), followed by another simple layer that produces the final output using a linear activation function.

[1]https://github.com/mmihaltz/word2vec-GoogleNews-vectors

The entire network is trained end-to-end through back-propagation in combination with the Adam optimization algorithm, using the great circle distance (i.e., average distance between the predicted coordinates and the real ones) as the loss function. In order to control overfitting and improve the generalization capabilities of the model, we use dropout regularization with a probability of 0.05 between each layer of the proposed neural network architecture. Dropout regularization is a simple procedure based on randomly dropping units, along whit their connections, from the neural network during training. Each unit is dropped with a fixed probability p independent of other units, effectively preventing the network units from co-adapting too much (Srivastava *et al.* 2014).

4. Experimental Evaluation

This section describes the datasets used for experiments, presents the experimental setup, and discusses the obtained results. Deep neural network architectures, such as the one proposed in this article, have successfully been applied to difficult challenges involving modeling sequences of text, such as language modeling, machine translation, or even measuring semantic similarity. Through experiments, we aimed to assess if indeed one such model could improve performance over state-of-the-art toponym resolution techniques. We used a Python library named Keras[1] for implementing the neural network architecture introduced in Section 3, and then leveraged some previous published datasets and a Wikipedia dataset, to comparatively evaluate its performance.

It is important to notice that although the deep learning procedure involves many parameters that can influence its performance, no significant effort was put into fine-tuning for optimal performance. The dimensionality of the outputs of the GRUs was set to 250, and dropout was considered as a regularization procedure after the last internal layer of the model, with a value of 0.1. The Adam optimization method (Kingma and Ba 2015) was used with the default value of 0.001 for the learning rate, a fuzz factor of 1e-08, and the β_1 and β_2 parameters respectively set to 0.9 and 0.999 (i.e., the default values in the Keras library).

All tests were performed on a standard PC with an Intel Core I7 6700 CPU running at 3.4 GHZ and an NVIDIA GeForce GTX 980 GPU, and with 16GB of RAM.

4.1. *Datasets And Experimental Methodology*

Our experiments relied on multiple datasets, each one consisting in annotated sentences containing toponyms with their respective geographic coordinates. The datasets used in the experiments were the following:

- **TR-CoNLL**: The TR-CoNLL corpus (Leidner 2008) contains 946 REUTERS news articles published in August 1996. Since this is a relatively small dataset, we performed three-fold cross-validation to avoid overfiting.
- **WOTR**: The WarOfTheRebellion (DeLozier *et al.* 2016) corpus consists in geoannotated data from the War Of The Rebelion (a large set of American Civial War archives). We used the same training and test sets as DeLozier *et al.* (2016).
- **Local-Glocal Lexicon (LGL):**A news articles corpus developed by Lieberman *et al.* (2010a) to evaluate toponym resolution systems on geographically localized text do-

[1] `https://keras.io`

	WOTR		TR-CoNLL		LGL		Wikipedia	
	Train	Test	Train	Test	Train	Test	Train	Test
Words	242,754	44,725	107,002	53,501	96,366	10,707	12,482,520	2,495,628
Toponyms	10,280	1,899	2,867	1,433	4,017	446	600,000	100,000
Unique Toponyms	1,952	712	608	424	1,138	297	108,965	34,729

Table 1.: Detailed statistics of the datasets

main. It contains 588 articles from sources selected to highlight less dominant place-names, (e.g. some articles are from Paris News a regional newspaper from Texas). We performed ten-fold cross-validation for the same reason as in TR-CoNLL.

- **Wikipedia dataset**: We also created a dataset by extracting every hyperlink corresponding to a geolocated Wikipedia article. Then, for every such hyperlink in an article, we extract the surrounding 24 words and associate that sentence with the coordinates of the article corresponding to the hyperlink. The training portion for this dataset contains 500,000 sentences and the test portion 100,000 sentences, randomly sampled from the entire dataset. We used this corpus to verify if by feeding more training instances to the network, we could improve the model's performance on the other datasets.

A detailed statistical characterization of the previous datasets is presented in Table 1.

Most previous works rely on precision and recall to evaluate the performance of their methods. Since we are using regression to predict the geographic locations of toponyms and not classification techniques, such as, choosing a candidate location from a gazetteer (Leidner 2008) or predicting a cell in a grid (Speriosu and Baldridge 2013, DeLozier *et al.* 2015), we cannot use these metrics to compare with state-of-the-art works. We instead rely on mean error distance, median error distance and A@161 (Leidner 2008, Eisenstein *et al.* 2010, Wing and Baldridge 2011, Santos *et al.* 2014), which correspond to the percentage of predicted distances less or equal than 161 km. The distances are calculated using Vincenty's formulae (Vincenty 1975), an iterative method that computes the geographical distance between two given points with an accuracy within 0.5 mm on the Earth's ellipsoid.

We performed three sets of experiments, differentiated by the training and testing data that was fed to the model.

4.2. *The Obtained Results*

Tables 2 and 3 present the obtained experimental results, effectively comparing the deep learning method proposed in Section 3 and state-of-the-art methods presented in Section 2.2 and 2.3.

In Table 2, **RNN-Geocoding** refers to the deep learning model trained and evaluated exclusively with data from the same corpus, whereas **RNN-Geocoding**$_{Wikipedia}$ refers to the experiments performed on the model trained with data from the English Wikipedia and then adjusted with data from corpus the method is being evaluated on.

The results presented on Table 2 show that, our method **RNN-Geocoding**,has evident different performances depending on which dataset it is being evaluated. For instance, when evaluating on the WOTR dataset, the deep learning method has the best overall performances in terms of mean distance, whereas with LGL and TR-CONNL it performs poorly. The difference between WOTR and the other datasets used in the experiments, is that almost every toponyms are *compressed* in the United States area. So even if a test instance is not seen in training, is very likely that the model will return a prediction that

	WOTR			TR-CoNLL			LGL		
Method	Mean	Median	A@161	Mean	Median	A@161	Mean	Median	A@161
RANDOM	2216	-	22.2	3891	1523.9	38.4	2852	1078	26.1
POPULATION	1483	-	63.1	219	30.3	90.5	1529	38	62.7
TopoCluster	482	-	57.0	560	122	53.2	1735	274	25.5
TopoClusterGaz	468	-	**71.5**	**209**	**0.0**	**93.2**	1540	**1.5**	61.4
WISTR	895	-	65.5	281	30.5	89.1	1264	27	64.0
WISTR+SPIDER	489	-	67.0	432	30.7	87.4	830	3	**77.7**
TRIPDL	848	**0.0**	60.2	1488	37.0	72.9	1311	46	60.9
TRAWL	944	**0.0**	70.8	237	30.5	89.7	2072	324	46.9
GeoSem	445	-	68.0	-	-	-	-	-	-
RNN-Geocoding	**173**	113	67.0	1225	291	25.0	1480	596	12.0
RNN-Geocoding$_{Wikipedia}$	230	154	53.0	727	172	46.0	**657**	252	30.3

Table 2.: Experimental results obtained by the proposed method, compared with state-of-the-art results

Dataset	Mean	Median	A@161
Wikipedia	1253	437	14.0
Wikipedia/WOTR	3572	2186	4.0
Wikipedia/TR-CoNLL	2167	406	21.9
Wikipedia/LGL	3427	1718	9.7

Table 3.: Experimental results obtained by the proposed method when trained with Wikipedia data and evaluated on other datasets.

belongs to the United States territory, thus the error distance will not be significantly high, as the result of the A@161 metric shows (67%). In the case of TR-CoNNL and LGL datasets, the toponyms are very dispersed, which means that is more likely for our method to make really bad predictions. This is evident by looking at the difference between the mean and the median performances on WOTR and the same metrics on the other datasets. One can infer by this results that the mean metric on TR-CoNLL and LGL is being inflated by bad predictions more than in WOTR.

When we feed Wikipedia training data to the model, we get a significantly increase in performance on TR-CoNLL and especially LGL, where we get the best overall mean error distance. Since we are feeding geo-referenced data from all over the world, we see a decrease in performance on the WOTR corpus, because the model can now make wrong predictions outside the United States. Despite this, we still get the second best result in terms of mean error distance.

Table 3 presents the results obtained from evaluating the method trained solely on data from the Wikipedia. The performance on the different metrics, show that Wikipedia data alone, is not enough to train a model that is capable to behave acceptably on data from different corpora. In some cases, the wrong predictions can be explained by inconsistent labeling on the evaluating corpora. For instance, the worst prediction by the pre-trained model with Wikipedia data when evaluated on TR-CoNLL is the toponym *New Zealand*. The predicted coordinates are actually close to the country's location, but since the toponym has a wrong label on TR-CoNLL, corresponding to a place in *Spain*, when evaluating, the model outputs an error equivalent to the distance between the prediction and that point in *Spain*.

5. Conclusions and Future Work

This article presented a novel method, based on a deep neural network architecture, for addressing the task of toponym resolution. Taking inspiration on previous research focused on natural language inference, the proposed neural network leverages recurrent units for encoding three input sequences, latter processing these representations in order to predict geographic coordinates. Using previous works datasets and data extracted from Wikipedia, we showed that the proposed method can, in some cases, outperform state-of-the-art results.

Despite the interesting results there are also many ideas for future work, since different options can be considered for improving the neural network architecture. Our neural architecture leverages GRUs to encode sequences of words, but other types of recurrent nodes have also recently been proposed. For instance, the Minimal Gated Unit approach Zhou *et al.* (2016), Heck and Salem (2017) relies on a simplified model with just a single gate. Having less parameters to train can contribute to improving the model effectiveness. In contrast, Multi-Function Recurrent Units (Mu-FuRUs) adopt an elaborate gating mechanism that allows for additional differentiable functions as composition operations, leading to models that can better capture the nuances involved in encoding word sequences (Weissenborn and Rocktäschel 2016). Other alternatives include Long Short-Term Memory (LSTM) networks with coupled gates (Greff *et al.* 2015), Structurally Constrained Recurrent Networks (Mikolov *et al.* 2014), IRNNs (Le *et al.* 2015), and many other LSTM or GRU variants (Greff *et al.* 2015, Jozefowicz *et al.* 2015).

Recent studies had relevant results with the usage of attention mechanisms on deep neural networks for document classification (Shimaoka *et al.* 2016, Yang *et al.* 2016). It would be interesting to apply these techniques on the task of toponym resolution.

A more general approach would be using deep learning for toponym resolution, but address the problem as a classification task instead of regression, in a similar faction as Speriosu and Baldridge (2013) and DeLozier *et al.* (2015).

References

Anastácio, I., *et al.*, 2011. Supervised Learning for Linking Named Entities to Knowledge Base Entries.. *In*: *Proceedings of the Text Analysis Conference*.

Ardanuy, M.C. and Sporleder, C., 2017. Toponym Disambiguation in Historical Documents Using Semantic and Geographic Features. *In*: *Proceedings of the International Conference on Digital Access to Textual Cultural Heritage*.

Bowman, S.R., *et al.*, 2015. A Large Annotated Corpus for Learning Natural Language Inference. *In*: *Proceedings of the Conference on Empirical Methods in Natural Language Processing*.

Breiman, L., 2001. Random forests. *Machine learning*, 45 (1).

Burges, C.J.C., 2010. From Ranknet to Lambdarank to LambdamMAR: An overview. *Learning*, 11.

Chung, J., *et al.*, 2014. Empirical Evaluation of Gated Recurrent Neural Networks on Sequence Modeling. *arXiv preprint arXiv:1412.3555*.

DeLozier, G., Baldridge, J., and London, L., 2015. Gazetteer-Independent Toponym Resolution Using Geographic Word Profiles. *In*: *Proceedings of the Conference of the Association for the Advancement of Artificial Intelligence*.

DeLozier, G., *et al.*, 2016. Creating a Novel Geolocation Corpus from Historical Texts.

In: *Proceedings of the Linguistic Annotation Workshop of the Association for Computational Linguistics.*

Eisenstein, J., *et al.*, 2010. A Latent Variable Model for Geographic Lexical Variation. *In*: *Proceedings of the Conference on Empirical Methods in Natural Language Processing.*

Goodfellow, I., Bengio, Y., and Courville, A., 2016. *Deep Learning.* MIT Press.

Greff, K., *et al.*, 2015. LSTM: A Search Space Odyssey. *arXiv preprint arXiv:1503.04069.*

Gregory, I., *et al.*, 2015. Geoparsing, GIS, and Textual Analysis: Current Developments in Spatial Humanities Research. *International Journal of Humanities and Arts Computing*, 9 (1).

Grover, C., *et al.*, 2010. Use of the Edinburgh Geoparser for Georeferencing Digitized Historical Collections.. *Philosophical transactions. Series A, Mathematical, physical, and engineering sciences*, 368 (1925).

Han, X., Sun, L., and Zhao, J., 2011. Collective Entity Linking in Web Text: A Graph-based Method. *In*: *Proceedings of the International Association for Computing Machinery SIGIR Conference on Research and Development in Information Retrieval.*

Heck, J. and Salem, F.M., 2017. Simplified Minimal Gated Unit Variations for Recurrent Neural Networks. *arXiv preprint arXiv:1701.03452.*

Ji, H. and Grishman, R., 2011. Knowledge Base Population: Successful Approaches and Challenges. *In*: *Proceedings of the Annual Meeting of the Association for Computational Linguistics: Human Language Technologies.*

Jozefowicz, R., Zaremba, W., and Sutskever, I., 2015. An Empirical Exploration of Recurrent Network Architectures. *In*: *Proceedings of the International Conference on Machine Learning.*

Kingma, D. and Ba, J., 2015. Adam: A Method for Stochastic Optimization. *In*: *Proceedings of the International Conference for Learning Representations.*

Kullback, S. and Leibler, R.A., 1951. On Information and Sufficiency. *The annals of mathematical statistics*, 22 (1).

Le, Q.V., Jaitly, N., and Hinton, G.E., 2015. A Simple Way to Initialize Recurrent Networks of Rectified Linear Units. *arXiv preprint arXiv:1504.00941.*

LeCun, Y., Bengio, Y., and Hinton, G., 2015. Deep learning. *Nature*, 521 (7553).

Leidner, J.L., 2007. Toponym Resolution in Text: Annotation, Evaluation and Applications of Spatial Grounding. *In*: *Proceedings of Association for Computing Machinery.*

Leidner, J.L., 2008. Toponym Resolution in Text: Annotation, Evaluation and Applications of Spatial Grounding of Place Names. *School of Informatics University of Edinburgh.*

Lieberman, M.D. and Samet, H., 2012. Adaptive Context Features for Toponym Resolution in Streaming News. *In*: *Proceedings of the International Association for Computing Machinery SIGIR Conference on Research and Development in Information Retrieval.*

Lieberman, M.D., Samet, H., and Sankaranarayanan, J., 2010a. Geotagging with Local Lexicons to Build Indexes for Textually-specified Spatial Data. *In*: *Proceedings of the International Conference on Data Engineering.*

Lieberman, M.D., Samet, H., and Sankaranayananan, J., 2010b. Geotagging: Using Proximity, Sibling, and Prominence Clues to Understand Comma Groups. *In*: *Proceedings of the Workshop on Geographic Information Retrieval.*

Liu, Y., *et al.*, 2016. Learning Natural Language Inference using Bidirectional LSTM model and Inner-Attention. *arXiv preprint arXiv:1605.09090.*

Mikolov, T., *et al.*, 2014. Learning Longer Memory in Recurrent Neural Networks. *arXiv preprint arXiv:1412.7753.*

Mueller, J. and Thyagarajan, A., 2016. Siamese Recurrent Architectures for Learning Sentence Similarity. *In*: *Proceedings of the Association for the Advancement of Artificial Intelligence*.

Ord, J.K. and Getis, A., 1995. Local Spatial Autocorrelation Statistics: Distributional Issues and an Application. *Geographical analysis*, 27 (4).

Rocktäschel, T., *et al.*, 2016. Reasoning About Entailment with Neural Attention. *In*: *Proceedings of the International Conference on Learning Representations*.

Rumelhart, D.E., Hinton, G.E., and Williams, R.J., 1988. Learning Representations by Back-propagating Errors. *Cognitive modeling*, 5 (3).

Santos, J., Anastácio, I., and Martins, B., 2014. Using machine learning methods for disambiguating place references in textual documents. *GeoJournal*, 80 (3).

Santos, R., Murrieta-Flores, P., and Martins, B., 2017. Learning to Combine Multiple String Similarity Metrics for Effective Toponym Matching. *(currently under review)*.

Schmidhuber, J., 2015. Deep Learning in Neural Networks: An Overview. *Neural Networks*, 61 (1).

Schuster, M. and Paliwal, K.K., 1997. Bidirectional Recurrent Neural Networks. *IEEE Transactions on Signal Processing*, 45 (11).

Shen, W., Wang, J., and Han, J., 2015. Entity Linking with a Knowledge Base: Issues, Techniques, and Solutions. *IEEE Transactions on Knowledge and Data Engineering*, 27 (2).

Shimaoka, S., *et al.*, 2016. Neural Architectures for Fine-grained Entity Type Classification. *arXiv preprint arXiv:1606.01341*.

Speriosu, M. and Baldridge, J., 2013. Text-Driven Toponym Resolution using Indirect Supervision. *In*: *Proceedings of the Annual Meeting of the Association for Computational Linguistics*.

Srivastava, N., *et al.*, 2014. Dropout: A Simple Way to Prevent Neural Networks from Overfitting. *Journal of Machine Learning Research*, 15 (1).

Vincenty, T., 1975. Direct and Inverse Solutions of Geodesics on the Ellipsoid with Application of Nested Equations. *Survey Review*, 22 (176).

Wan, S., *et al.*, 2016. A Deep Architecture for Semantic Matching with Multiple Positional Sentence Representations. *In*: *Proceedings of the Association for the Advancement of Artificial Intelligence*.

Weissenborn, D. and Rocktäschel, T., 2016. MuFuRU: The Multi-Function Recurrent Unit. *In*: *Proceedings of the Association for Computational Linguistics Workshop on Representation Learning for Natural Language Processing*.

Wing, B.P. and Baldridge, J., 2011. Simple Supervised Document Geolocation with Geodesic Grids. *In*: *Proceedings of the Annual Meeting of the Association for Computational Linguistics*.

Yang, Z., *et al.*, 2016. Hierarchical Attention Networks for Document Classification.. *In*: *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistic*.

Yin, W. and Schütze, H., 2015. Convolutional neural network for paraphrase identification. *In*: *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics*.

Zhou, G.B., *et al.*, 2016. Minimal Gated Unit for Recurrent Neural Networks. *International Journal of Automation and Computing*, 13 (3).