Which Melbourne? Augmenting Geocoding with Maps

Milan Gritta, Mohammad Taher Pilehvar and Nigel Collier

Language Technology Lab
Department of Theoretical and Applied Linguistics
University of Cambridge

{mg711,mp792,nhc30}@cam.ac.uk

Abstract

The purpose of text geolocation is to associate geographic information contained in a document with a set (or sets) of coordinates, either *implicitly* by using linguistic features and/or *explicitly* by using geographic metadata combined with heuristics. We introduce a geocoder (location mention disambiguator) that achieves state-of-the-art (SOTA) results on three diverse datasets by exploiting the implicit lexical clues. Moreover, we propose a new method for systematic encoding of geographic metadata to generate two *distinct* views of the same text. To that end, we introduce the *Map Vector* (*MapVec*), a sparse representation obtained by plotting prior geographic probabilities, derived from population figures, on a World Map. We then integrate the implicit (language) and explicit (map) features to significantly improve a range of metrics. We also introduce an open-source dataset for geoparsing of news events covering global disease outbreaks and epidemics to help future evaluation in geoparsing.

1 Introduction

Geocoding[1] is a specific case of text geolocation, which aims at disambiguating place references in text. For example, *Melbourne* can refer to more than ten possible locations and a geocoder's task is to identify the place coordinates for the intended *Melbourne* in a context such as "*Melbourne* hosts one of the four annual Grand Slam tennis tournaments." This is central to the success of tasks such as indexing and searching documents by geography (Bhargava et al., 2017), geospatial analysis of social media (Buchel and Pennington, 2017), mapping of disease risk using integrated data (Hay et al., 2013), and emergency response systems (Ashktorab et al., 2014). Previous geocoding methods (Section 2) have leveraged lexical semantics to associate the implicit geographic information in natural language with coordinates. These models have achieved good results in the past. However, focusing *only* on lexical features, to the exclusion of other feature spaces such as the Cartesian Coordinate System, puts a ceiling on the amount of semantics we are able to extract from text. Our proposed solution is the *Map Vector* (*MapVec*), a sparse, geographic vector for explicit modelling of geographic distributions of location mentions. As in previous work, we use population data and geographic coordinates, observing that the most populous *Melbourne* is also the most likely to be the intended location. However, MapVec is the first instance, to our best knowledge, of the topological semantics of context locations explicitly isolated into a standardized vector representation, which can then be easily transferred to an independent task and combined with other features. MapVec is able to encode the prior geographic distribution of any number of locations into a single vector. Our extensive evaluation shows how this representation of context locations can be integrated with linguistic features to achieve a significant improvement over a SOTA lexical model. MapVec can be deployed as a standalone neural geocoder, significantly beating the population baseline, while remaining effective with simpler machine learning algorithms.

This paper's contributions are: (1) *Lexical Geocoder* outperforming existing systems by analysing only the textual context; (2) *MapVec*, a geographic representation of locations using a sparse, probabilistic vector to extract and isolate spatial features; (3) *CamCoder*, a novel geocoder

[1] Also called Toponym Resolution in related literature.

that exploits both lexical and geographic knowledge producing SOTA results across multiple datasets; and (4) *GeoVirus*, an open-source dataset for the evaluation of geoparsing (Location Recognition *and* Disambiguation) of news events covering global disease outbreaks and epidemics.

2 Background

Depending on the task objective, geocoding methodologies can be divided into two *distinct* categories: (1) *document geocoding*, which aims at locating a piece of text as a whole, for example geolocating Twitter users (Rahimi et al., 2016, 2017; Roller et al., 2012; Rahimi et al., 2015), Wikipedia articles and/or web pages (Cheng et al., 2010; Backstrom et al., 2010; Wing and Baldridge, 2011; Dredze et al., 2013; Wing and Baldridge, 2014). This is an active area of NLP research (Hulden et al., 2015; Melo and Martins, 2017, 2015; Iso et al., 2017); (2) *geocoding of place mentions*, which focuses on the disambiguation of location (named) entities i.e. this paper and (Karimzadeh et al., 2013; Tobin et al., 2010; Grover et al., 2010; DeLozier et al., 2015; Santos et al., 2015; Speriosu and Baldridge, 2013; Zhang and Gelernter, 2014). Due to the differences in evaluation and objective, the categories cannot be directly or fairly compared. Geocoding is typically the second step in *Geoparsing*. The first step, usually referred to as *Geotagging*, is a Named Entity Recognition component which extracts all location references in a given text. This phase may optionally include metonymy resolution, see (Zhang and Gelernter, 2015; Gritta et al., 2017a). The goal of geocoding is to choose the correct coordinates for a location mention from a set of candidates. Gritta et al. (2017b) provided a comprehensive survey of five recent geoparsers. The authors established an evaluation framework, with a new dataset, for their experimental analysis. We use this evaluation framework in our experiments. We briefly describe the methodology of each geocoder featured in our evaluation (names are capitalised and appear in italics) as well as survey the related work in geocoding.

Computational methods in geocoding broadly divide into rule-based, statistical and machine learning-based. *Edinburgh Geoparser* (Tobin et al., 2010; Grover et al., 2010) is a fully rule-based geocoder that uses hand-built heuristics combined with large lists from Wikipedia and the Geonames[2] gazetteer. It uses metadata (feature type, population, country code) with heuristics such as contextual information, spatial clustering and user locality to rank candidates. *GeoTxT* (Karimzadeh et al., 2013) is another rule-based geocoder with a free web service[3] for identifying locations in unstructured text and grounding them to coordinates. Disambiguation is driven by multiple heuristics and uses the administrative level (country, province, city), population size, the Levenshtein Distance of the place referenced and the candidate's name and spatial minimisation to resolve ambiguous locations. (Dredze et al., 2013) is a rule-based Twitter geocoder using only metadata (coordinates in tweets, GPS tags, user's reported location) and custom place lists for fast and simple geocoding. *CLAVIN* (Cartographic Location And Vicinity INdexer)[4] is an open-source geocoder, which offers context-based entity recognition and linking. It seems to be mostly rule-based though details of its algorithm are underspecified, short of reading the source code. Unlike the Edinburgh Parser, this geocoder seems to overly rely on population data, seemingly mirroring the behaviour of a naive population baseline. Rule-based systems can perform well though the variance in performance is high (see Table 1). *Yahoo! Placemaker* is a free web service with a proprietary geo-database and algorithm from Yahoo![5] letting anyone geoparse text in a globally-aware and language-independent manner. It is unclear how geocoding is performed, however, the inclusion of proprietary methods makes evaluation broader and more informative.

The statistical geocoder *Topocluster* (DeLozier et al., 2015) divides the world surface into a grid (0.5 x 0.5 degrees, approximately 60K tiles) and uses lexical features to model the geographic distribution of context words over this grid. Building on the work of Speriosu and Baldridge (2013), it uses a window of 15 words (our approach scales this up by more than 20 times) to perform hot spot analysis using Getis-Ord Local Statistic of individual words' association with geographic space. The classification decision was made by finding the grid square with the strongest overlap of

[2] http://www.geonames.org/
[3] http://www.geotxt.org/
[4] https://clavin.bericotechnologies.com
[5] https://developer.yahoo.com/geo/

individual geo-distributions. Hulden et al. (2015) used Kernel Density Estimation to learn the word distribution over a world grid with a resolution of 0.5 x 0.5 degrees and classified documents with Kullback-Leibler divergence or a Naive Bayes model, reminiscent of an earlier approach by Wing and Baldridge (2011). Roller et al. (2012) used the Good-Turing Frequency Estimation to learn document probability distributions over the vocabulary with Kullback-Leibler divergence as the similarity function to choose the correct bucket in the k-d tree (world representation). Iso et al. (2017) combined Gaussian Density Estimation with a CNN-model to geolocate Japanese tweets with Convolutional Mixture Density Networks.

Among the recent machine learning methods, bag-of-words representations combined with a Support Vector Machine (Melo and Martins, 2015) or Logistic Regression (Wing and Baldridge, 2014) have also achieved good results. For Twitter-based geolocation (Zhang and Gelernter, 2014), bag-of-words classifiers were successfully augmented with social network data (Jurgens et al., 2015; Rahimi et al., 2016, 2015). The machine learning-based geocoder by *Santos et al. (2015)* supplemented lexical features, represented as a bag-of-words, with an exhaustive set of manually generated geographic features and spatial heuristics such as geospatial containment and geodesic distances between entities. The ranking of locations was learned with LambdaMART (Burges, 2010). Unlike our geocoder, the addition of geographic features did not significantly improve scores, reporting: "The geo-specific features seem to have a limited impact over a strong baseline system." Unable to obtain a codebase, their results feature in Table 1. The latest neural network approaches (Rahimi et al., 2017) with normalised bag-of-word representations have achieved SOTA scores when augmented with social network data for Twitter document (user's concatenated tweets) geolocation (Bakerman et al., 2018).

3 Methodology

Figure 1 shows our new geocoder *CamCoder* implemented in Keras (Chollet, 2015). The lexical part of the geocoder has three inputs, from the top: Context Words (location mentions excluded), Location Mentions (context words excluded) and the Target Entity (up to 15 words long) to be

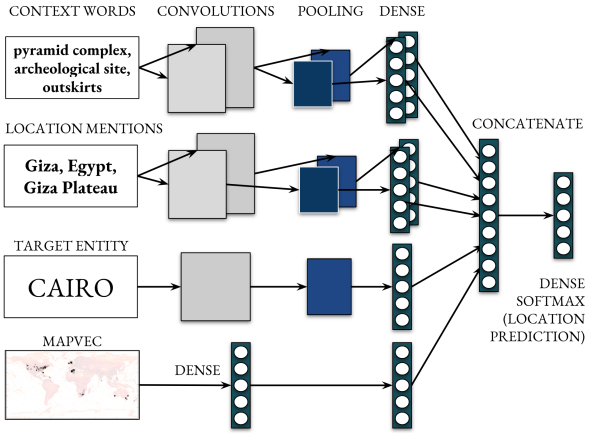

Figure 1: The *CamCoder* neural architecture. It is possible to split CamCoder into a *Lexical* (top 3 inputs) model and a *MapVec* model (see Table 2).

geocoded. Consider an example disambiguation of *Cairo* in a sentence: "*The Giza pyramid complex is an archaeological site on the Giza Plateau, on the outskirts of Cairo, Egypt.*". Here, *Cairo* is the Target Entity; *Egypt*, *Giza* and *Giza Plateau* are the Location Mentions; the rest of the sentence forms the Context Words (excluding stopwords). The *context window* is up to 200 words each side of the Target Entity, approximately an order of magnitude larger than most previous approaches.

We used separate layers, convolutional and/or dense (fully-connected), with ReLu activations (Nair and Hinton, 2010) to break up the task into smaller, focused modules in order to learn distinct lexical feature patterns, phrases and keywords for different types of inputs, concatenating only at a higher level of abstraction. Unigrams and bigrams were learned for context words and location mentions (1,000 filters of size 1 and 2 for each input), trigrams for the target entity (1,000 filters of size 3). Convolutional Neural Networks (CNNs) with Global Maximum Pooling were chosen for their position invariance (detecting location-indicative words anywhere in context) and efficient input size scaling. The dense layers have 250 units each, with a dropout layer ($p = 0.5$) to prevent overfitting. The fourth input is MapVec, the geographic vector representation of location mentions. It feeds into two dense layers with 5,000 and 1,000 units respectively. The concatenated hidden layers then get fully connected to the softmax layer. The model is optimised with RMSProp (Tieleman and Hinton, 2012). We approach geocoding as a classification task where the model predicts one of

7,823 classes (units in the softmax layer in Figure 1), each being a 2x2 degree tile representing part of the world's surface, slightly coarser than MapVec (see Section 3.1 next). The coordinates of the location candidate with the smallest *FD* (Equation 1) are the model's final output.

$$FD = error - error \frac{candidatePop}{maximumPop} Bias \quad (1)$$

FD for each candidate is computed by reducing the prediction *error* (the distance from predicted coordinates to candidate coordinates) by the value of *error* multiplied by the estimated prior probability (candidate population divided by maximum population) multiplied by the *Bias* parameter. The value of $Bias = 0.9$ was determined to be optimal for highest development data scores and is *identical for all* highly diverse test datasets. Equation 1 is designed to bias the model towards more populated locations to reflect real-world data.

3.1 The Map Vector (MapVec)

Word embeddings and/or distributional vectors encode a word's meaning in terms of its *linguistic* context. However, location (named) entities also carry explicit *topological semantic knowledge* such as a coordinate position and a population count for all places with an identical name. Until now, this knowledge was only used as part of simple disparate heuristics and manual disambiguation procedures. However, it is possible to plot this spatial data on a world map, which can then be reshaped into a 1D *feature vector*, or a *Map Vector*, the geographic representation of location mentions. MapVec is a novel standardised method for generating geographic features from text documents *beyond* lexical features. This enables a strong geocoding classification performance gain by extracting additional spatial knowledge that would normally be ignored. Geographic semantics cannot be inferred from language alone (too imprecise and incomplete). Word embeddings and distributional vectors use language/words as an *implicit* container of geographic information. Map Vector uses a low-resolution, probabilistic world map as an *explicit* container of geographic information, giving us two types of semantic features from the same text. In related papers on the generation of location representations, Rahimi et al. (2017) inverted the task of geocoding Twitter users to predict word

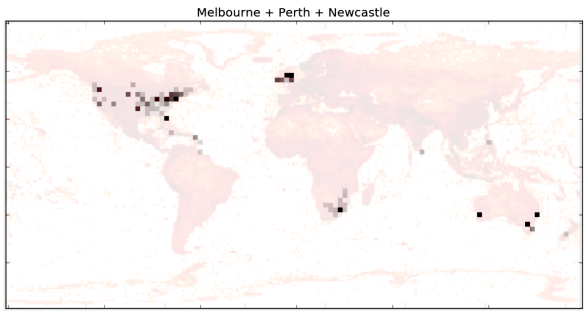

Figure 2: MapVec visualisation (before reshaping into a 1D vector) for *Melbourne*, *Perth* and *Newcastle*, showing their combined prior geographic probabilities. Darker tiles have higher probability.

probability from a set of coordinates. A continuous representation of a *region* was generated by using the hidden layer of the neural network. However, all locations in the same region will be assigned an identical vector, which assumes that their semantics are also identical. Another way to obtain geographic representations is by generating embeddings directly from Geonames data using heuristics-driven DeepWalk (Perozzi et al., 2014) with geodesic distances (Kejriwal and Szekely, 2017). However, to assign a vector, places must first be disambiguated (catch-22). While these generation methods are original and interesting in theory, deploying them in the real-world is infeasible, hence we invented the Map Vector.

MapVec initially begins as a 180x360 world map of geodesic tiles. There are other ways of representing the surface of the Earth such as using nested hierarchies (Melo and Martins, 2015) or k-dimensional trees (Roller et al., 2012), however, this is beyond the scope of this work. The 1x1 tile size, in degrees of geographic coordinates, was empirically determined to be optimal to keep MapVec's size computationally efficient while maintaining meaningful resolution. This map is then populated with the *prior geographic distribution* of each location mentioned in context (see Figure 2 for an example). We use population count to *estimate* a location's prior probability as more populous places are more likely to be mentioned in common discourse. For each location mention and for each of its ambiguous candidates, their prior probability is added to the correct tile indicating its geographic position (see Algorithm 1). Tiles that cover areas of open water (64.1%) were removed to reduce size. Finally,

```
Data: Text ← article, paragraph, tweet, etc.
Result: MapVec location(s) representation
Locs ← extractLocations(Text);
MapVec ← new array(length=23,002);
for each l in Locs do
    Cands ← queryCandidatesFromDB(l);
    maxPop ← maxPopulationOf(Cands);
    for each c in Cands do
        prior ← populationOf(c) / maxPop;
        i ← coordinatesToIndex(c);
        MapVec[i] ← MapVec[i] + prior;
    end
end
m ← max(MapVec);
return MapVec / m;
```

Algorithm 1: MapVec generation. For each extracted location $\boldsymbol{l}$ in *Locs*, estimate the prior probability of each candidate $\boldsymbol{c}$. Add $\boldsymbol{c}$'s prior probability to the appropriate array position at index $\boldsymbol{i}$ representing its geographic position/tile. Finally, normalise the array (to a $[0-1]$ range) by dividing by the maximum value of the MapVec array.

this world map is reshaped into a one-dimensional *Map Vector* of length 23,002.

The following features of MapVec are the most salient: *Interpretability:* Word vectors typically need intrinsic (Gerz et al., 2016) and extrinsic tasks (Senel et al., 2017) to interpret their semantics. MapVec generation is a fully transparent, human readable and modifiable method. *Efficiency:* MapVec is an efficient way of embedding *any number* of locations using the same standardised vector. The alternative means creating, storing, disambiguating and computing with millions of unique location vectors. *Domain Independence:* Word vectors vary depending on the source, time, type and language of the training data and the parameters of generation. MapVec is language-independent and stable over time, domain, size of dataset since the world geography is objectively measured and changes very slowly.

3.2 Data and Preprocessing

Training data was generated from geographically annotated Wikipedia pages (dumped February 2017). Each page provided up to 30 training instances, limited to avoid bias from large pages. This resulted in collecting approximately 1.4M training instances, which were uniformly subsampled down to 400K to shorten training cycles as further increases offer diminishing returns. We used the Python-based NLP toolkit Spacy[6] (Honnibal and Johnson, 2015) for text preprocessing. All words were lowercased, lemmatised, any stopwords, dates, numbers and so on were replaced with a special token ("0"). Word vectors were initialised with pretrained word embeddings[7] (Pennington et al., 2014). We do not employ explicit feature selection as in (Bo et al., 2012), only a minimum frequency count, which was shown to work almost as well as deliberate selection (Van Laere et al., 2014). The vocabulary size was limited to the most frequent 331K words, minimum ten occurrences for words and two for location references in the 1.4M training corpus. A final training instance comprises four types of context information: Context Words (excluding location mentions, up to 2x200 words), Location Mentions (excluding context words, up to 2x200 words), Target Entity (up to 15 words) and the MapVec geographic representation of context locations. We have also checked for any overlaps between our Wikipedia-based training data and the WikToR dataset. Those examples were removed. The aforementioned 1.4M Wikipedia training corpus was once again uniformly subsampled to generate a *disjoint* development set of 400K instances. While developing our models mainly on this data, we also used small subsets of LGL (18%), GeoVirus (26%) and WikToR (9%) described in Section 4.2 to verify that development set improvements generalised to target domains.

4 Evaluation

Our evaluation compares the geocoding performance of six systems from Section 2, our geocoder (CamCoder) and the population baseline. Among these, our CNN-based model is the only neural approach. We have included all open-source/free geocoders *in working order* we were able to find and they are the most up-to-date versions. Tables 1 and 2 feature several machine learning algorithms including Long-Short Term Memory (LSTM) (Hochreiter and Schmidhuber, 1997) to reproduce context2vec (Melamud et al., 2016), Naive Bayes (Zhang, 2004) and Random Forest (Breiman, 2001) using three diverse datasets.

[6] https://spacy.io/
[7] https://nlp.stanford.edu/

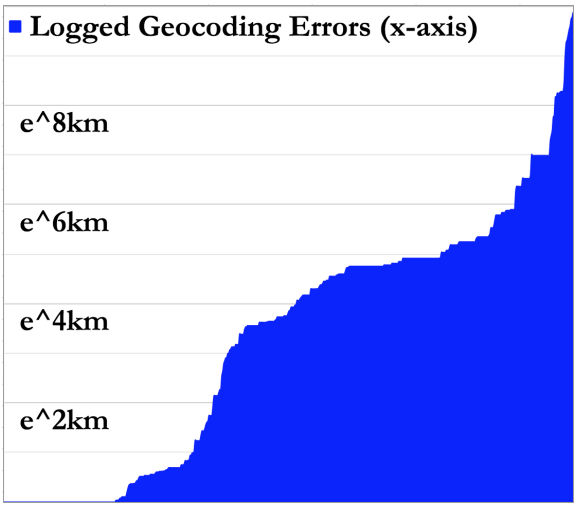

Figure 3: The AUC (range $[0-1]$) is calculated using the Trapezoidal Rule. Smaller errors mean a smaller (blue) area, which means a lower score and therefore better geocoding results.

4.1 Geocoding Metrics

We use the three standard and comprehensive metrics, each measuring an important aspect of geocoding, giving an accurate, holistic evaluation of performance. A more detailed cost-benefit analysis of geocoding metrics is available in (Karimzadeh, 2016) and (Gritta et al., 2017b). (1) *Average (Mean) Error* is the sum of all geocoding errors per dataset divided by the number of errors. It is an informative metric as it also indicates the total error but treats all errors as equivalent and is sensitive to outliers; (2) *Accuracy@161km* is the percentage of errors that are smaller than 161km (100 miles). While it is easy to interpret, giving fast and intuitive understanding of geocoding performance in percentage terms, it ignores all errors greater than 161km; (3) *Area Under the Curve* (*AUC*) is a comprehensive metric, initially introduced for geocoding in (Jurgens et al., 2015). AUC reduces the importance of large errors (1,000km+) since accuracy on successfully resolved places is more desirable. While it is not an intuitive metric, AUC is robust to outliers and measures *all* errors. A versatile geocoder should be able to maximise all three metrics.

4.2 Evaluation Datasets

News Corpus: The Local Global Lexicon (LGL) by Lieberman et al. (2010) contains 588 news articles (4460 test instances), which were collected from geographically distributed newspaper sites. This is the most frequently used geocoding evaluation dataset to date. The toponyms are mostly smaller places no larger than a US state. Approximately 16% of locations in the corpus do not have any coordinates assigned; hence, we do not use those in the evaluation, which is also how the previous figures were obtained. *Wikipedia Corpus*: This corpus was deliberately designed for ambiguity hence the population heuristic is not effective. Wikipedia Toponym Retrieval (WikToR) by Gritta et al. (2017b) is a programmatically created corpus and although not necessarily representative of the real world distribution, it is a test of ambiguity for geocoders. It is also a large corpus (25,000+ examples) containing the first few paragraphs of 5,000 Wikipedia pages. High quality, free and open datasets are not readily available (GeoVirus tries to address this). The following corpora could not be included: WoTR (DeLozier et al., 2016) due to limited coverage (southern US) and domain type (historical language, the 1860s), (De Oliveira et al., 2017) contains fewer than 180 locations, GeoCorpora (Wallgrün et al., 2017) could not be retrieved in full due to deleted Twitter users/tweets, GeoText (Eisenstein et al., 2010) only allows for user geocoding, SpatialML (Mani et al., 2010) involves prohibitive costs, GeoSemCor (Buscaldi and Rosso, 2008) was annotated with WordNet senses (rather than coordinates).

4.3 GeoVirus: a New Test Dataset

We now introduce GeoVirus, an open-source test dataset for the evaluation of geoparsing of news events covering global disease outbreaks and epidemics. It was constructed from free WikiNews[8] and collected during 08/2017 - 09/2017. The dataset is suitable for the evaluation of Geotagging/Named Entity Recognition and Geocoding/Toponym Resolution. Articles were identified using the WikiNews search box and keywords such as Ebola, Bird Flu, Swine Flu, AIDS, Mad Cow Disease, West Nile Disease, etc. Off-topic articles were not included. Buildings, POIs, street names and rivers were not annotated.

Annotation Process. (1) The WikiNews contributor(s) who wrote the article annotated most, but not all location references. The first author checked those annotations and identified further references, then proceeded to extract the place name, indices of the start and end characters in

[8] https://en.wikinews.org

Geocoder	Area Under Curve†			Average Error‡			Accuracy@161km		
	LGL	WIK	GEO	LGL	WIK	GEO	LGL	WIK	GEO
CamCoder	**22 (18)**	**33 (37)**	**31 (32)**	7 **(5)**	**11 (9)**	**3 (3)**	**76 (83)**	**65 (57)**	**82 (80)**
Edinburgh	25 (22)	53 (58)	33 (34)	8 (8)	31 (30)	5 (4)	**76** (80)	42 (36)	78 (78)
Yahoo!	34 (35)	44 (53)	40 (44)	**6 (5)**	23 (25)	**3 (3)**	72 (75)	52 (39)	70 (65)
Population	27 (22)	68 (71)	32 **(32)**	12 (10)	45 (42)	5 **(3)**	70 (79)	22 (14)	80 **(80)**
CLAVIN	26 (20)	70 (69)	32 (33)	13 (9)	43 (39)	6 (5)	71 (80)	16 (16)	79 **(80)**
GeoTxt	29 (21)	70 (71)	33 (34)	14 (9)	47 (45)	6 (5)	68 (80)	18 (14)	79 (79)
Topocluster	38 (36)	63 (66)	NA	12 (8)	38 (35)	NA	63 (71)	26 (20)	NA
Santos et al.	NA	NA	NA	8	NA	NA	71	NA	NA

Table 1: Results on LGL, WikToR (WIK) and GeoVirus (GEO). Lower AUC and Average Error are better while higher Acc@161km is better. Figures in *brackets* are scores on identical subsets of each dataset. †Only the AUC decimal part shown. ‡Average Error rounded up to the nearest 100km.

text, assigned coordinates and the Wikipedia page URL for each location. (2) A second pass over the entire dataset by the first author to check and/or remedy annotations. (3) A computer program checked that locations were tagged correctly, checking coordinates against the Geonames Database, URL correctness, eliminating any duplicates and validating XML formatting. Places without a Wikipedia page (0.6%) were assigned Geonames coordinates. (4) The second author annotated a random 10% sample to obtain an Inter-Annotator Agreement, which was 100% for geocoding and an F-Score of 92.3 for geotagging. *GeoVirus in Numbers:* Annotated locations: 2,167, Unique: 685, Continents: 94, Number of articles: 229, Most frequent places (21% of total): US, Canada, China, California, UK, Mexico, Kenya, Africa, Australia, Indonesia; Mean location occurrence: 3.2, Total word count: 63,205.

5 Results

All tested models (except CamCoder) operate as end-to-end systems; therefore, it is not possible to perform geocoding separately. Each system geoparses its particular majority of the dataset to obtain a representative data sample, shown in Table 1 as strongly correlated scores for subsets of different sizes, with which to assess model performance. Table 1 also shows scores in *brackets* for the overlapping partition of all systems in order to compare performance on identical instances: GeoVirus 601 (26%), LGL 787 (17%) and WikToR 2,202 (9%). The geocoding difficulty based on the ambiguity of each dataset is: LGL (moderate to hard), WIK (very hard), GEO (easy to moderate). A population baseline also features in the evaluation. The baseline is conceptually simple: choose the candidate with the highest population, akin to the most frequent word sense in WSD. Table 1 shows the effectiveness of this heuristic, which is competitive with many geocoders, even *outperforming* some. However, the baseline is not effective on WikToR as the dataset was deliberately constructed as a tough ambiguity test. Table 1 shows how several geocoders mirror the behaviour of the population baseline. This simple but effective heuristic is rarely used in system comparisons, and where evaluated (Santos et al., 2015; Leidner, 2008), it is inconsistent with expected figures (due to unpublished resources, we are unable to investigate).

We note that no single computational paradigm dominates Table 1. The rule-based (Edinburgh, GeoTxt, CLAVIN), statistical (Topocluster), machine learning (CamCoder, Santos) and other (Yahoo!, Population) geocoders occupy different ranks across the three datasets. Due to space constraints, Table 1 does not show figures for another type of scenario we tested, a shorter lexical context, using 200 words instead of the standard 400. CamCoder proved to be robust to reduced context, with only a small performance decline. Using the same format as Table 1, AUC errors for LGL increased from 22 (18) to 23 (19), WIK from 33 (37) to 37 (40) and GEO remained the same at 31 (32). This means that reducing model input size to save computational resources would still deliver accurate results. Our CNN-based lexical model performs at SOTA levels (Table 2) proving the effectiveness of linguistic features while being

Geocoder	System configuration			Dataset			Average
	Language Features	+	MapVec Features	LGL	WIK	GEO	
CamCoder	CNN		MLP	**0.22**	**0.33**	**0.31**	**0.29**
Lexical Only	CNN		–	0.23	0.39	0.33	0.32
MapVec Only	–		MLP	0.25	0.41	0.32	0.33
Context2vec†	LSTM		MLP	0.24	0.38	0.33	0.32
Context2vec	LSTM		–	0.27	0.47	0.39	0.38
Random Forest	MapVec features only, no lexical input			0.26	0.36	0.33	0.32
Naive Bayes	MapVec features only, no lexical input			0.28	0.56	0.36	0.40
Population	–		–	0.27	0.68	0.32	0.42

Table 2: AUC scores for *CamCoder* and its *Lexical* and *MapVec* components (model ablation). Lower AUC scores are better. †Standard context2vec model augmented with MapVec representation.

the outstanding geocoder on the highly ambiguous WikToR data. The Multi-Layer Perceptron (MLP) model using only MapVec with no lexical features is almost as effective but more importantly, it is significantly better than the population baseline (Table 2). This is because the Map Vector benefits from wide contextual awareness, encoded in Algorithm 1, while a simple population baseline does not. When we combined the lexical *and* geographic feature spaces in one model (CamCoder[9]), we observed a substantial increase in the SOTA scores. We have also reproduced the *context2vec* model to obtain a continuous context representation using bidirectional LSTMs to encode lexical features, denoted as LSTM[10] in Table 2. This enabled us to test the effect of integrating MapVec into another deep learning model as opposed to CNNs. Supplemented with MapVec, we observed a significant improvement, demonstrating how enriching various neural models with a geographic vector representation boosts classification results.

Deep learning is the dominant paradigm in our experiments. However, it is important that MapVec is still effective with simpler machine learning algorithms. To that end, we have evaluated it with the Random Forest *without* using any lexical features. This model was well suited to the geocoding task despite training with only half of the 400K training data (due to memory constraints, partial fit is unavailable for batch training in SciKit Learn). Scores were on par with more sophisticated systems. The Naive Bayes was less effective with MapVec though still somewhat viable as a geocoder given the lack of lexical features and a naive algorithm, narrowly beating population. GeoVirus scores remain highly competitive across most geocoders. This is due to the nature of the dataset; locations skewed towards their dominant "senses" simulating ideal geocoding conditions, enabling high accuracy for the population baseline. GeoVirus alone may not serve as the best scenario to assess a geocoder's performance, however, it is nevertheless important and valuable to determine behaviour in a standard environment. For example, GeoVirus helped us diagnose Yahoo! Placemaker's lower accuracy in what should be an easy test for a geocoder. The figures show that while the average error is low, the accuracy@161km is noticeably lower than most systems. When coupled with other complementary datasets such as LGL and WikToR, it facilitates a comprehensive assessment of geocoding behaviour in many types of scenarios, exposing potential domain dependence. We note that GeoVirus has a dual function, NER (not evaluated but useful for future work) and Geocoding. We made all of our resources freely available[11] for full reproducibility (Goodman et al., 2016).

5.1 Discussion and Errors

The Pearson correlation coefficient of the target entity *ambiguity* and the *error size* was only $r \approx 0.2$ suggesting that CamCoder's geocoding errors do not simply rise with location ambiguity. Errors were also not correlated ($r \approx 0.0$) with population size with all types of locations geocoded to various degrees of accuracy. All error curves follow

[9] Single model settings/parameters for all tests.

[10] https://keras.io/layers/recurrent/

[11] https://github.com/milangritta/

a power law distribution with between 89% and 96% of errors less than 1500km, the rest rapidly increasing into thousands of kilometers. Errors also appear to be uniformly geographically distributed across the world. The strong lexical component shown in Table 2 is reflected by the lack of a relationship between *error size* and the *number of locations* found in the context. The *number of total words* in context is also independent of geocoding accuracy. This suggests that CamCoder learns strong linguistic cues beyond simple association of place names with the target entity and is able to cope with flexible-sized contexts. A CNN Geocoder would expect to perform well for the following reasons: Our context window is 400 words rather than 10-40 words as in previous approaches. The model learns 1,000 feature maps per input and per feature type, tracking 5,000 different word patterns (unigrams, bigrams and trigrams), a significant text processing capability. The lexical model also takes advantage of our own 50-dimensional word embeddings, tuned on geographic Wikipedia pages only, allowing for greater generalisation than bag-of-unigrams models; and the large training/development datasets (400K each), optimising geocoding over a diverse global set of places allowing our model to generalise to unseen instances. We note that MapVec generation is sensitive to NER performance with higher F-Scores leading to better quality of the geographic vector representation(s). Precision errors can introduce noise while recall errors may withhold important locations. The average F-Score for the featured geoparsers is $F \approx 0.7$ (standard deviation ≈ 0.1). Spacy's NER performance over the three datasets is also $F \approx 0.7$ with a similar variation between datasets. In order to further interpret scores in Tables 1 and 2, with respect to maximising geocoding performance, we briefly discuss the *Oracle* score. Oracle is the geocoding performance upper bound given by the Geonames data, i.e. the *highest possible score(s)* using Geonames coordinates as the geocoding output. In other words, it quantifies the *minimum error* for each dataset given the *perfect* location disambiguation. This means it quantifies the difference between "gold standard" coordinates and the coordinates in the Geonames database. The following are the Oracle scores for LGL (*AUC=0.04, a@161km=99*) annotated with Geonames, WikToR (*AUC=0.14, a@161km=92*) and GeoVirus (*AUC=0.27, a@161km=88*), which are annotated with Wikipedia data. Subtracting the Oracle score from a geocoder's score quantifies the scope of its *theoretical future improvement*, given a particular database/gazetteer.

6 Conclusions and Future Work

Geocoding methods commonly employ lexical features, which have proved to be very effective. Our lexical model was the best language-only geocoder in extensive tests. It is possible, however, to go *beyond* lexical semantics. Locations also have a rich topological meaning, which has not yet been successfully isolated and deployed. We need a means of extracting and encoding this additional knowledge. To that end, we introduced MapVec, an algorithm and a container for encoding context locations in geodesic vector space. We showed how CamCoder, using lexical *and* MapVec features, outperformed both approaches, achieving a new SOTA. MapVec remains effective with various machine learning frameworks (Random Forest, CNN and MLP) and substantially improves accuracy when combined with other neural models (LSTMs). Finally, we introduced GeoVirus, an open-source dataset that helps facilitate geoparsing evaluation across more diverse domains with different lexical-geographic distributions (Flatow et al., 2015; Dredze et al., 2016). Tasks that could benefit from our methods include social media placing tasks (Choi et al., 2014), inferring user location on Twitter (Zheng et al., 2017), geolocation of images based on descriptions (Serdyukov et al., 2009) and detecting/analyzing incidents from social media (Berlingerio et al., 2013). Future work may see our methods applied to *document* geolocation to assess the effectiveness of scaling geodesic vectors from paragraphs to entire documents.

Acknowledgements

We gratefully acknowledge the funding support of the Natural Environment Research Council (NERC) PhD Studentship NE/M009009/1 (Milan Gritta, DREAM CDT), EPSRC (Nigel Collier) Grant Number EP/M005089/1 and MRC (Mohammad Taher Pilehvar) Grant Number MR/M025160/1 for PheneBank. We also gratefully acknowledge NVIDIA Corporation's donation of the Titan Xp GPU used for this research.

References

Zahra Ashktorab, Christopher Brown, Manojit Nandi, and Aron Culotta. 2014. Tweedr: Mining twitter to inform disaster response. In *ISCRAM*.

Lars Backstrom, Eric Sun, and Cameron Marlow. 2010. Find me if you can: improving geographical prediction with social and spatial proximity. In *Proceedings of the 19th international conference on World wide web*. ACM, pages 61–70.

Jordan Bakerman, Karl Pazdernik, Alyson Wilson, Geoffrey Fairchild, and Rian Bahran. 2018. Twitter geolocation: A hybrid approach. *ACM Transactions on Knowledge Discovery from Data (TKDD)* 12(3):34.

Michele Berlingerio, Francesco Calabrese, Giusy Di Lorenzo, Xiaowen Dong, Yiannis Gkoufas, and Dimitrios Mavroeidis. 2013. Safercity: a system for detecting and analyzing incidents from social media. In *Data Mining Workshops (ICDMW), 2013 IEEE 13th International Conference on*. IEEE, pages 1077–1080.

Preeti Bhargava, Nemanja Spasojevic, and Guoning Hu. 2017. Lithium nlp: A system for rich information extraction from noisy user generated text on social media. *arXiv preprint arXiv:1707.04244* .

Han Bo, Paul Cook, and Timothy Baldwin. 2012. Geolocation prediction in social media data by finding location indicative words. In *Proceedings of COLING*. pages 1045–1062.

Leo Breiman. 2001. Random forests. *Machine learning* 45(1):5–32.

Olga Buchel and Diane Pennington. 2017. Geospatial analysis. *The SAGE Handbook of Social Media Research Methods* pages 285–303.

Chris J.C. Burges. 2010. From ranknet to lambdarank to lambdamart: An overview. Technical report. https://www.microsoft.com/en-us/research/publication/from-ranknet-to-lambdarank-to-lambdamart-an-overview/.

Davide Buscaldi and Paulo Rosso. 2008. A conceptual density-based approach for the disambiguation of toponyms. *International Journal of Geographical Information Science* 22(3):301–313.

Zhiyuan Cheng, James Caverlee, and Kyumin Lee. 2010. You are where you tweet: a content-based approach to geo-locating twitter users. In *Proceedings of the 19th ACM international conference on Information and knowledge management*. ACM, pages 759–768.

Jaeyoung Choi, Bart Thomee, Gerald Friedland, Liangliang Cao, Karl Ni, Damian Borth, Benjamin Elizalde, Luke Gottlieb, Carmen Carrano, Roger Pearce, et al. 2014. The placing task: A large-scale geo-estimation challenge for social-media videos and images. In *Proceedings of the 3rd ACM Multimedia Workshop on Geotagging and Its Applications in Multimedia*. ACM, pages 27–31.

François Chollet. 2015. Keras. https://github.com/fchollet/keras.

Maxwell Guimarães De Oliveira, Cláudio de Souza Baptista, Cláudio EC Campelo, and Michela Bertolotto. 2017. A gold-standard social media corpus for urban issues. In *Proceedings of the Symposium on Applied Computing*. ACM, pages 1011–1016.

Grant DeLozier, Jason Baldridge, and Loretta London. 2015. Gazetteer-independent toponym resolution using geographic word profiles. In *AAAI*. pages 2382–2388.

Grant DeLozier, Ben Wing, Jason Baldridge, and Scott Nesbit. 2016. Creating a novel geolocation corpus from historical texts. *LAW X* page 188.

Mark Dredze, Miles Osborne, and Prabhanjan Kambadur. 2016. Geolocation for twitter: Timing matters. In *HLT-NAACL*. pages 1064–1069.

Mark Dredze, Michael J Paul, Shane Bergsma, and Hieu Tran. 2013. Carmen: A twitter geolocation system with applications to public health. In *AAAI workshop on expanding the boundaries of health informatics using AI (HIAI)*. volume 23, page 45.

Jacob Eisenstein, Brendan O'Connor, Noah A Smith, and Eric P Xing. 2010. A latent variable model for geographic lexical variation. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, pages 1277–1287.

David Flatow, Mor Naaman, Ke Eddie Xie, Yana Volkovich, and Yaron Kanza. 2015. On the accuracy of hyper-local geotagging of social media content. In *Proceedings of the Eighth ACM International Conference on Web Search and Data Mining*. ACM, pages 127–136.

Daniela Gerz, Ivan Vulić, Felix Hill, Roi Reichart, and Anna Korhonen. 2016. Simverb-3500: A large-scale evaluation set of verb similarity. *arXiv preprint arXiv:1608.00869* .

Steven N Goodman, Daniele Fanelli, and John PA Ioannidis. 2016. What does research reproducibility mean? *Science translational medicine* 8(341):341ps12–341ps12.

Milan Gritta, Mohammad Taher Pilehvar, Nut Limsopatham, and Nigel Collier. 2017a. Vancouver welcomes you! minimalist location metonymy resolution. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. volume 1, pages 1248–1259.

Milan Gritta, Mohammad Taher Pilehvar, Nut Limsopatham, and Nigel Collier. 2017b. Whats missing in geographical parsing? .

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 of London A: Mathematical, Physical and Engineering Sciences* 368(1925):3875–3889.

Simon I Hay, Katherine E Battle, David M Pigott, David L Smith, Catherine L Moyes, Samir Bhatt, John S Brownstein, Nigel Collier, Monica F Myers, Dylan B George, et al. 2013. Global mapping of infectious disease. *Phil. Trans. R. Soc. B* 368(1614):20120250.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation* 9(8):1735–1780.

Matthew Honnibal and Mark Johnson. 2015. An improved non-monotonic transition system for dependency parsing. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, Lisbon, Portugal, pages 1373–1378. https://aclweb.org/anthology/D/D15/D15-1162.

Mans Hulden, Miikka Silfverberg, and Jerid Francom. 2015. Kernel density estimation for text-based geolocation. In *AAAI*. pages 145–150.

Hayate Iso, Shoko Wakamiya, and Eiji Aramaki. 2017. Density estimation for geolocation via convolutional mixture density network. *arXiv preprint arXiv:1705.02750* .

David Jurgens, Tyler Finethy, James McCorriston, Yi Tian Xu, and Derek Ruths. 2015. Geolocation prediction in twitter using social networks: A critical analysis and review of current practice. *ICWSM* 15:188–197.

Morteza Karimzadeh. 2016. Performance evaluation measures for toponym resolution. In *Proceedings of the 10th Workshop on Geographic Information Retrieval*. ACM, page 8.

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

Mayank Kejriwal and Pedro Szekely. 2017. Neural embeddings for populated geonames locations. In *International Semantic Web Conference*. Springer, pages 139–146.

Jochen L Leidner. 2008. *Toponym resolution in text: Annotation, evaluation and applications of spatial grounding of place names*. Universal-Publishers.

Michael D Lieberman, Hanan Samet, and Jagan Sankaranarayanan. 2010. Geotagging with local lexicons to build indexes for textually-specified spatial data. In *2010 IEEE 26th International Conference on Data Engineering (ICDE 2010)*. IEEE, pages 201–212.

Inderjeet Mani, Christy Doran, Dave Harris, Janet Hitzeman, Rob Quimby, Justin Richer, Ben Wellner, Scott Mardis, and Seamus Clancy. 2010. Spatialml: annotation scheme, resources, and evaluation. *Language Resources and Evaluation* 44(3):263–280.

Oren Melamud, Jacob Goldberger, and Ido Dagan. 2016. context2vec: Learning generic context embedding with bidirectional lstm. In *CoNLL*. pages 51–61.

Fernando Melo and Bruno Martins. 2015. Geocoding textual documents through the usage of hierarchical classifiers. In *Proceedings of the 9th Workshop on Geographic Information Retrieval*. ACM, page 7.

Fernando Melo and Bruno Martins. 2017. Automated geocoding of textual documents: A survey of current approaches. *Transactions in GIS* 21(1):3–38.

Vinod Nair and Geoffrey E Hinton. 2010. Rectified linear units improve restricted boltzmann machines. In *Proceedings of the 27th international conference on machine learning (ICML-10)*. pages 807–814.

Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014. Glove: Global vectors for word representation. In *Empirical Methods in Natural Language Processing (EMNLP)*. pages 1532–1543. http://www.aclweb.org/anthology/D14-1162.

Bryan Perozzi, Rami Al-Rfou, and Steven Skiena. 2014. Deepwalk: Online learning of social representations. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, pages 701–710.

Afshin Rahimi, Timothy Baldwin, and Trevor Cohn. 2017. Continuous representation of location for geolocation and lexical dialectology using mixture density networks. *arXiv preprint arXiv:1708.04358* .

Afshin Rahimi, Trevor Cohn, and Timothy Baldwin. 2016. pigeo: A python geotagging tool .

Afshin Rahimi, Duy Vu, Trevor Cohn, and Timothy Baldwin. 2015. Exploiting text and network context for geolocation of social media users. *arXiv preprint arXiv:1506.04803* .

Stephen Roller, Michael Speriosu, Sarat Rallapalli, Benjamin Wing, and Jason Baldridge. 2012. Supervised text-based geolocation using language models on an adaptive grid. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*. Association for Computational Linguistics, pages 1500–1510.

João Santos, Ivo Anastácio, and Bruno Martins. 2015. Using machine learning methods for disambiguating place references in textual documents. *GeoJournal* 80(3):375–392.

Lutfi Kerem Senel, Ihsan Utlu, Veysel Yucesoy, Aykut Koc, and Tolga Cukur. 2017. Semantic structure and interpretability of word embeddings. *arXiv preprint arXiv:1711.00331* .

Pavel Serdyukov, Vanessa Murdock, and Roelof Van Zwol. 2009. Placing flickr photos on a map. In *Proceedings of the 32nd international ACM SIGIR conference on Research and development in information retrieval*. ACM, pages 484–491.

Michael Speriosu and Jason Baldridge. 2013. Text-driven toponym resolution using indirect supervision. In *ACL (1)*. pages 1466–1476.

Tijmen Tieleman and Geoffrey Hinton. 2012. Lecture 6.5-rmsprop: Divide the gradient by a running average of its recent magnitude. *COURSERA: Neural networks for machine learning* 4(2):26–31.

Richard Tobin, Claire Grover, Kate Byrne, James Reid, and Jo Walsh. 2010. Evaluation of georeferencing. In *proceedings of the 6th workshop on geographic information retrieval*. ACM, page 7.

Olivier Van Laere, Jonathan Quinn, Steven Schockaert, and Bart Dhoedt. 2014. Spatially aware term selection for geotagging. *IEEE transactions on Knowledge and Data Engineering* 26(1):221–234.

Jan Oliver Wallgrün, Morteza Karimzadeh, Alan M MacEachren, and Scott Pezanowski. 2017. Geocorpora: building a corpus to test and train microblog geoparsers. *International Journal of Geographical Information Science* pages 1–29.

Benjamin Wing and Jason Baldridge. 2014. Hierarchical discriminative classification for text-based geolocation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. pages 336–348.

Benjamin P Wing and Jason Baldridge. 2011. Simple supervised document geolocation with geodesic grids. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies-Volume 1*. Association for Computational Linguistics, pages 955–964.

Harry Zhang. 2004. The optimality of naive bayes. *AA* 1(2):3.

Wei Zhang and Judith Gelernter. 2014. Geocoding location expressions in twitter messages: A preference learning method. *Journal of Spatial Information Science* 2014(9):37–70.

Wei Zhang and Judith Gelernter. 2015. Exploring metaphorical senses and word representations for identifying metonyms. *arXiv preprint arXiv:1508.04515* .

Xin Zheng, Jialong Han, and Aixin Sun. 2017. A survey of location prediction on twitter. *arXiv preprint arXiv:1705.03172* .