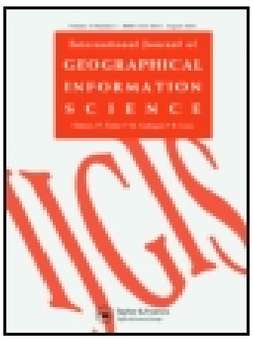

International Journal of Geographical Information Science

ISSN: (Print) (Online) Journal homepage: https://www.tandfonline.com/loi/tgis20

GazPNE: annotation-free deep learning for place name extraction from microblogs leveraging gazetteer and synthetic data by rules

Xuke Hu, Hussein S. Al-Olimat, Jens Kersten, Matti Wiegmann, Friederike Klan, Yeran Sun & Hongchao Fan

To cite this article: Xuke Hu, Hussein S. Al-Olimat, Jens Kersten, Matti Wiegmann, Friederike Klan, Yeran Sun & Hongchao Fan (2021): GazPNE: annotation-free deep learning for place name extraction from microblogs leveraging gazetteer and synthetic data by rules, International Journal of Geographical Information Science, DOI: 10.1080/13658816.2021.1947507

To link to this article: https://doi.org/10.1080/13658816.2021.1947507

Published online: 07 Jul 2021.

INTERNATIONAL JOURNAL OF GEOGRAPHICAL INFORMATION SCIENCE
https://doi.org/10.1080/13658816.2021.1947507

Taylor & Francis
Taylor & Francis Group

RESEARCH ARTICLE

GazPNE: annotation-free deep learning for place name extraction from microblogs leveraging gazetteer and synthetic data by rules

Xuke Hu[a], Hussein S. Al-Olimat[b], Jens Kersten[a], Matti Wiegmann[c], Friederike Klan[a], Yeran Sun[d] and Hongchao Fan[e]

[a]Institute of Data Science, German Aerospace Center (DLR), Jena, Germany; [b]AI & Data Science, Tempus Labs Inc, Chicago, IL, USA; [c]Computer Science Department, Bauhaus-Universität Weimar, Weimar, Germany; [d]Depertment of Geography, College of Science, Swansea University, Swansea, UK; [e]Department of Civil and Environmental Engineering, Norwegian University of Science and Technology, Trondheim, Norway

ABSTRACT

Extracting precise location information from microblogs is a crucial task in many applications, particularly in disaster response, revealing where damages are, where people need assistance, and where help can be found. A crucial prerequisite to location extraction is place name extraction. In this paper, we present GazPNE: a hybrid approach to place name extraction which fuses rules, gazetteers, and deep learning techniques without requiring any manually annotated data. The core of the approach is to learn the intrinsic characteristics of multi-word place names with deep learning from gazetteers. Specifically, GazPNE consists of a rule-based system to select n-grams from the microblogs that potentially contain place names, and a C-LSTM model that decides if the selected n-gram is a place name or not. The C-LSTM is trained on 388.1 million examples containing 6.8 million positive examples with US and Indian place names extracted from OpenStreetMap and 381.3 million negative examples synthesized by rules. We evaluate GazPNE against the SoTA on a manually annotated 4,500 tweet dataset which contains 9,026 place names from three foods: 2016 in Louisiana (US), 2016 in Houston (US), and 2015 in Chennai (India). GazPNE achieves SotA performance on the test data with an F1 of 0.84.

ARTICLE HISTORY
Received 29 September 2020
Accepted 21 June 2021

KEYWORDS
Place name extraction; gazetteer; OpenStreetMap; synthetic data; rule; microblog; deep learning

1. Introduction

Online social media platforms, especially microblog platforms such as Twitter and Weibo, are responsive to real-world events and are useful for gathering situational information in real-time (Tapia *et al*. 2013, Reuter and Kaufhold 2018). For example, Twitter is widely used by both the public and the government in disaster response, such as earthquakes, floods, fire, terrorist attacks, and civil unrest (Alexander 2014, Ozdikis *et al*. 2017, Yuan and Liu 2018). When an emergency event occurs, extracting location information from tweets is crucial for keeping people and authorities informed about exact affected areas, where people can receive fresh water and food, and the locations where people need rescue and

CONTACT Xuke Hu xuke.hu@dlr.de

medical assistance (Kar *et al*. 2018, Maneriker *et al*. 2019). However, this task is a challenge since geo-tagged tweets are sparse. According to Cheng *et al*. (2010), Morstatter *et al*. (2013), and Kumar *et al*. (2017), only 0.42%, 3.17%, and 7.90% of the total number of tweets contain geo-tags, respectively, and these rarely reflect the mentions' exact geolocations (Middleton *et al*. 2018). Thus, it is necessary to extract the location information from tweet texts (i.e. making it an extractive problem rather than a retrieval problem). This task is called location extraction and consists of two steps: (1) identifying references to the locations in a text, known as toponym recognition or place name extraction, and (2) assigning geographic coordinates to the identified place name, known as toponym resolution or geocoding. This study proposes a new state-of-the-art method for place name extraction.

Currently, the approaches for place name extraction from microblogs can be categorized into four groups: hand-crafted rules, gazetteers, statistical learning, and hybrid approaches. The approaches based on handcrafted rules use pattern and regex-matching, which in turn rely on cue words or orthographic features (Gelernter and Balaji 2013, Giridhar *et al*. 2015). Handcrafted rules are fragile and cannot yield a general place name extractor considering the dramatic variation of the writing styles and numerous grammar errors in microblog texts (Ritter *et al*. 2011). The approaches based on gazetteers (Sultanik and Fink 2012, Middleton *et al*. 2018) extract place names by matching a sequence of tokens with the place names in a gazetteer. However, since gazetteers like OpenStreetMap (OSM) are incomplete and since place names are often mentioned in many colloquial forms (Beall 2010), approaches based on gazetteer matching frequently miss results, especially in microblogs (Al-Olimat *et al*. 2018). The approaches based on statistical learning (Finkel *et al*. 2005, Kumar and Singh 2019) extract place names mainly according to their contextual features. Examples are Stanza (Qi *et al*. 2020) and NeuroTPR (Wang *et al*. 2020). However, statistical learning requires many annotated sentences, which makes those approaches ineffective in most practical situations (Guerini *et al*. 2018). The hybrid approaches (Li and Sun 2014, Weissenbacher *et al*. 2015, Al-Olimat *et al*. 2018, Dutt *et al*. 2018) fuse gazetteers with rules or statistical learning to compensate for each other's weaknesses. However, previous hybrid approaches did not solve the incompleteness of gazetteers, place name variation or faced the challenge of sparsely annotated data.

This study proposes a novel hybrid approach that combines gazetteers, rules, and deep learning to yield a robust place name extractor, named GazPNE (**Gaz**etteer-enhanced **P**lace **N**ame **E**xtractor). The core of the approach is to learn the intrinsic characteristics of multi-word place names such that it can mitigate the incompleteness of gazetteers and place name variation issues. GazPNE shares some features with recent zero-shot learning work (Xie *et al*. 2016) since it does not require any annotated sentences in training procedures. Specifically, we obtain positive examples from OpenStreetMap. We then apply several rules to synthesize negative examples from the positive examples and the vocabulary in Google's pre-trained Word2Vec model (Mikolov *et al*. 2013b) and Glove-embeddings (Pennington *et al*. 2014). After this, a classification model is trained based on C-LSTM (Zhou *et al*. 2015), which fuses Convolutional Neural Networks (CNN) and Long Short-Term Memory (LSTM). Then, a simple part-of-speech (POS) rule is utilized to select valid n-grams in a microblog text. This is mainly to deal with hard examples for the model, such as 'me' and 'us', which could refer to persons or the abbreviation of Maine state and

the United States, respectively, depending on POS tags. Specifically, we use CMU ARK Twitter tagger (Gimpel *et al*. 2010) to determine the POS tag of each token of the micro-blog text. Finally, we apply the classifier on the valid n-grams and select the top non-overlapping candidates as the final set of detected place names. The main contribution of the study is that we propose a robust place name extraction approach, fusing deep learning and rules (expert knowledge), which can extract place names from micro-blogs at both coarse (e.g. country and city) and fine-grained levels (e.g. street, creek, and public buildings). It leverages only the data extracted from OpenStreetMap without the need for any manually annotated data. The approach has the potential of being generalized to regions worldwide since OpenStreetMap is available worldwide.

The remainder of this paper is structured as follows: In Section 2, we conduct a review of related works. We present the workflow of the proposed approach and give the details of each component in Section 3. We evaluate the proposed approach in Section 4. We then discuss some key issues and limitations of the study in Section 5, and Section 6 concludes the paper.

2. Related works

The approaches for place name extraction can be coarsely divided into four groups. (1) Rule-based approaches, (2) Gazetteer-matching-based approaches, (3) Statistical Named Entity Recognition-based approaches, (4) Hybrid approaches.

Rule-based approaches: Place names in texts have certain characteristics, which could be represented by rules. For instance, Gelernter and Balaji (2013) proposed a street and building parser for Spanish and English texts by using POS rules, such as adjective plus noun and street and building indicator words, such as 'calle', 'cl', 'carreterra', 'cr', 'cra'. In each rule, the string must satisfy the POS rule, and also the last word must be building or street indicator words. Giridhar *et al*. (2015) used road-traffic-related tweets to detect and locate point events, such as car accidents. As for event localization, raw tweets are first tokenized and then tagged using a POS tagger. A set of grammar rules were defined according to the composition of nouns, determiners, adjectives, cardinal numbers, conjunctions, and possessive endings. Moreover, to decrease false positives, an additional grammar-based rule was implemented based on true location-identifiers, which were commonly preceded by prepositions, such as *in, around, between*, and *after*. Generally, the rule-based approach is simple and high efficient in computation. In some cases, it can achieve a promising tagging result. However, it is challenging to define complete rules that can cover all the cases of place names in tweet texts, which is also troublesome. Thus, it is likely that the defined rules work well on one test data but fail on the other test data.

Gazetteer search and matching: Compared with rule-based approaches, gazetteer-based approaches implemented as entity matching requires less manual effort. Especially when free and rich geospatial gazetteers are available, a high tagging accuracy can be usually achieved. A gazetteer is the dictionary of geospatial places with names and geo-coordinates. Popular geospatial gazetteers include OpenStreetMap, GeoNames, and the geo-tagged Wikipedia database. A few heuristics and stop words are normally used to limit the set of candidate place names. The valid ones can then be extracted by matching the token sequence of the tweet text with the items in the gazetteer. That can find not only the valid place names but also specify

their geo-coordinates. Many previous studies (Sultanik and Fink 2012, Middleton *et al*. 2013, 2018) have used gazetteers to extract place names. For instance, Sultanik and Fink (2012) proposed RapidGeo, which can efficiently perform a 'fuzzy' matching between strings, mapping toponyms to likely locations in gazetteers. In this way, it can mitigate misspellings in the input data. Specifically, it hashes the gazetteer's toponyms based on a phonetic encoding algorithm that roughly encodes the way a word would be pronounced by an Anglophone. The data structure maps each distinct phonetic encoding of the toponyms in the gazetteer to a K-D Tree data structure, which contains the associated toponyms in the gazetteer indexed by their location values. Based on the K-D Tree, the nearest neighbor search can be efficiently performed. Middleton *et al*. (2018) proposed a multilingual place name extractor based on OpenStreetMap. A set of heuristics are applied to each OpenStreetMap location name to perform location name expansion and create a set of n-gram location phrases. Then, unigram location names that are non□nouns usually result in false positives are filtered using a multilingual WordNet corpus lookup. Gazetteer-based approaches can achieve high tagging accuracy on the test data where most of the place names are included in gazetteers. However, it is also common that many place names found in tweets are missing from gazetteers due to place name variation and incompleteness of gazetteers.

Statistical learning: Recently, many studies adopted statistical learning to extract place names (Gelernter and Mushegian 2011, Lingad *et al*. 2013, Unankard *et al*. 2015, Limsopatham and Collier 2016, Das and Purves 2019, Kumar and Singh 2019, Wang *et al*. 2020, Qi *et al*. 2020). Given abundant annotated data, statistical learning-based approaches can recognize place names according to context cues and intrinsic features of place names. Place name recognition is a subtask of name entity recognition (NER), which has been extensively investigated. Therefore, many studies (Gelernter and Mushegian 2011, Lingad *et al*. 2013, Unankard *et al*. 2015) leverage existing statistical-based NER models to extract place names from social media streams. For instance, Lingad *et al*. (2013) retrained Stanford NER (Finkel *et al*. 2005), which implements a linear chain Conditional Random Field (CRF) model to label sequences of words in a text into entity types (e.g. Person, Organization, and Location). The classical NER systems are implemented through traditional machine learning approaches, which require manual definition and selection of features according to expert knowledge.

Recently, some studies used deep learning to extract location names while avoiding feature engineering. Limsopatham and Collier (2016) proposed an approach for recognizing name entities from tweet texts by enabling bidirectional long short-term memory (Bi-LSTM) to automatically learn orthographic features without requiring feature engineering using both character embeddings and word embeddings. Kumar and Singh (2019) utilized a Convolutional Neural Network (CNN) based model for place name extraction. NeuroTPR (Wang *et al*. 2020) extended a general Bi-LSTM architecture with several features to account for linguistic irregularities in Twitter texts, such as the use of character embeddings to capture the morphological features of words, and POS tags and contextual embeddings to capture the semantics of tokens in tweets. The approach mitigates the effort for annotating a large training dataset by generating annotated data from Wikipedia articles for the task of location name extraction. Similar on-average performance was achieved by Stanza's contextualized string representation-based sequence tagger (Qi *et al*. 2020). They trained a forward and a backward character-level LSTM then

concatenate the output from them at tagging time with word embeddings fed into a one-layer Bi-LSTM with CRF-based decoder at prediction time.

Although deep learning shows promising NER performance (particularly in place name extraction), annotated data is very limiting, especially for novel and evolving events. To mitigate the challenge, Guerini *et al*. (2018) proposed a domain portable zero-shot learning approach for entity recognition, which does not assume any annotated sentences at training time. More specifically, they trained a 3-layer Bi-LSTM model based only on available dictionaries and synthesized examples. It was then applied to recognize new entities in user utterances. Through multiple experiments in two languages (i.e. English and Italian) and three different domains (i.e. furniture, food, clothing), the proposed approach outperformed several competitive baselines, with minimal requirements of linguistic features. This work inspired the idea of our study.

Hybrid approaches: Every single technique has its drawbacks. Thus, researchers have proposed fusing different techniques to achieve the best of all (Bontcheva *et al*. 2013, Li and Sun 2014, Weissenbacher *et al*. 2015, Al-Olimat *et al*. 2018, Dutt *et al*. 2018). For instance, Weissenbacher *et al*. (2015) proposed using a hybrid approach combining dictionary-based and rule-based heuristics for the detection and disambiguation of locations in articles. It first uses a built-in dictionary (GeoNames) to detect mentions of place names in articles. A black list and a set of rules were created to remove noisy entries found in GeoNames. It then disambiguates the place names using a distance heuristic, a population heuristic, and a novel heuristic utilizing knowledge from GenBank metadata. Similarly, Dutt *et al*. (2018) proposed inferring place names mentioned in tweets by fusing rules and gazetteers. They used a POS tagger to find the proper nouns based on heuristics and then used regular expression matching to mitigate the ambiguation of proper nouns with the prefix and suffix words that appear in the beginning or end of place names. Last, the list of extracted phrases is then verified using a gazetteer.

Apart from the fusion of rules and gazetteers, combining statistical learning and gazetteers was also investigated. For instance, Li and Sun (2014) proposed a novel solution named Petar, extracting fine-grained locations mentioned in tweets. They first construct a point of interest (POI) inventory or gazetteer from check-in data in Foursquare, which contains not only formal names of POI but also informal abbreviations. To deal with ambiguous and incomplete POI names in the inventory, a time-aware POI tagger based on Conditional Random Field (CRF) is further proposed. The tagger takes POI inventory as a knowledge base and utilizes four types of features (i.e. lexical, grammatical, geographical, and BILOU schema features) to label POI names and their temporal awareness. In this sense, the POI inventory can be considered as a noisy version of a gazetteer to the CRF classifier. Al-Olimat *et al*. (2018) proposed a Location Name Extraction tool (LNEx), which used n-gram statistics and location-related dictionaries to handle the abbreviations and automatically filter and augments the location names in gazetteers (handling name contractions and auxiliary contents). It can detect the boundaries of multi-word location names and thereby delimit them in texts.

In a nutshell, pure rule-based approaches are not generalizable and sometimes fragile. Gazetteer-based approaches cannot deal with incompleteness of gazetteers and variation of place names. Traditional statistical learning-based approaches face the challenge of data sparsity. Existing hybrid approaches normally combine gazetteers with statistical

learning or rules, but still cannot solve the incompleteness of gazetteers and variation of place names issues or face the challenge of sparsely annotated sentences.

3. GazPNE: place name extraction processing chain

Two aspects mainly inspire the idea of this study. **(1) The challenge of detecting unknown multi-word place names facing gazetteer-based approaches**. To recognize the short names (such as *'Emsley High School'*) of a full name (such as *'Emsley A Laney High School'*) in gazetteers, the gazetteer-based approach (Al-Olimat *et al*. 2018) needs to augment the gazetteer by extracting the short name of a full name and add it to the gazetteer. However, this can also generate incorrect place names, such as *'`The Apartments'* from *'The x Apartments'*, causing false positives. Moreover, many place names (e.g. *'Mississippi Coast church'* and *'emergency operation center'*) are missing in gazetteers (such as OpenStreetMap) but appear in tweet texts, rendering them unextractable. In a nutshell, one of the biggest challenges of gazetteer-based approaches lies in the recognition of multi-word place names, which normally own certain lexical and orthographic characteristics. This inspired us to use a deep learning model to learn the inherent characteristics of multi-word place names. By doing so, the learned model can recognize variants of multi-word place names as well as missing ones.

(2) Combination of statistical learning with rules (or expert knowledge). This study attempts to use only gazetteers to train a classification model without requiring manually annotated data, which is costly to collect on scale. Specifically, we obtain millions of positive examples (place names) from gazetteers and then manually define a couple of rules to synthesize as accurate negative examples as possible based on the positive ones (e.g. *'The University of Mississippi'*) and general words, such as to insert general words in the beginning and end of a place name (e.g. *'is The University of Mississippi at'*) or to reorder the tokens of a place name (e.g. *'The of Mississippi University'*). These heuristics might be imperfect and can lead to noisy examples. However, according to Rolnick *et al*. (2017), deep learning algorithms can tolerate modest amounts of noise in training data given abundant correct training examples. That provided us a new way of combining deep learning with rules (expert knowledge) by generating enough training examples through rules and meanwhile mitigating the interference of noisy examples by using deep learning (Rolnick *et al*. 2017).

The workflow of GazPNE is shown in Figure 1. The number in the box denotes the section numbering of each component. It consists of three main stages. The first is to generate training examples. Specifically, the gazetteer is first filtered and then augmented, such as by replacing a token (e.g. *'road'*) with its abbreviation (e.g. *'rd'*). Furthermore, we apply a set of rules to synthesize negative examples based on the positive examples. The second is to train a neural classifier based on the generated training examples. Specifically, we adopt an existing deep-learning architecture named C-LSTM (Zhou *et al*. 2015). The last stage is to apply the trained classifier to unseen microblog texts. Specifically, a microblog text is first preprocessed by tokenizing the text, tagging the POS of tokens, and selecting valid subsets by a simple POS rule. Then, the trained neural classifier is applied to select valid subset candidates and the top non-overlapping subsets with the highest positive probability are selected as detected place names. The novelty of

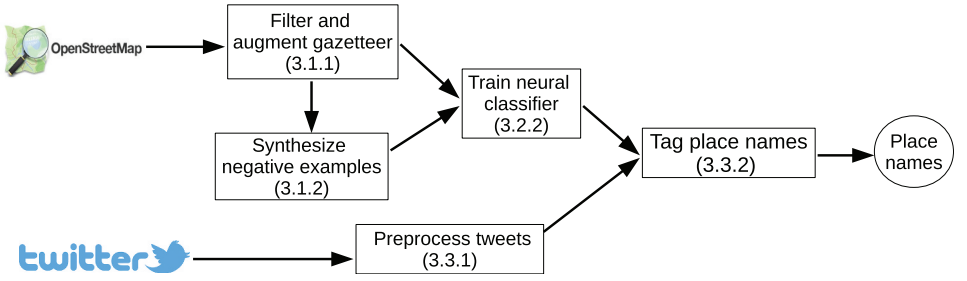

Figure 1. Workflow of our proposed place name extraction approach (GazPNE).

the proposed approach lies in the fusion of deep learning, gazetteers, and rules, which does not require any manually annotated data.

3.1. Training example generation

This step's key task is to produce as abundant and accurate training examples as possible from gazetteers and general words. The general words are obtained from well-known pre-trained word embeddings, including Glove (Pennington *et al*. 2014) and Word2Vec (Mikolov *et al*. 2013b) embeddings. From the Word2Vec embedding, the frequency of each word can also be obtained. Thus, a general word list can be produced by giving more frequent words more attention, such as copying a word multiple times according to its frequency. The general words include words, numbers, English alphabet letters (e.g. 'a'), and number-prefix-suffix words (e.g. 'ft','inch", and 'pm') that do not appear in gazetteers. The number-prefix-suffix words are extracted from the Glove and Word2Vec embedding vocabularies, which consist of numbers and words or alphabets (e.g. '10-ft'). To generalize the representation of numbers, the numbers 0 to 9 are replaced by '0'. That is, '8','12","238", and '1235' are converted to '0', '00', '000', and '0000', respectively. The token in positive examples, which consists of numbers, is broken into numbers and words, such as to break *'hwy00'* into *'hwy'* and *'00'*. Apart from general words, location category names (e.g. 'school', 'road', 'town') are also used to synthesize negative examples. They are extracted by counting the last word of the place names in gazetteers, and the one whose count is over a certain threshold is regarded as a location category name. Besides, all the characters are converted to lower case, and all non-English and non-number characters are removed.

3.1.1. Obtaining positive examples

(1) **Filtering**: To obtain high-quality positive examples, we define two rules to filter place names in OpenStreetMap. First, OpenStreetMap suffers from data quality issues. For instance, many tiny, unimportant, and unpopulated items are mapped or named on OpenStreetMap by volunteers themselves while the mainstream maps (i.e. Google Maps and Bing maps) normally do not represent or name them. Thus, we first remove the items whose attributes are 'highway = footway', 'highway = service', 'highway = cycleway', 'highway = track', 'highway = path', or 'place = locality'. For instance, Figure 2 shows two OpenStreetMap items incorrectly

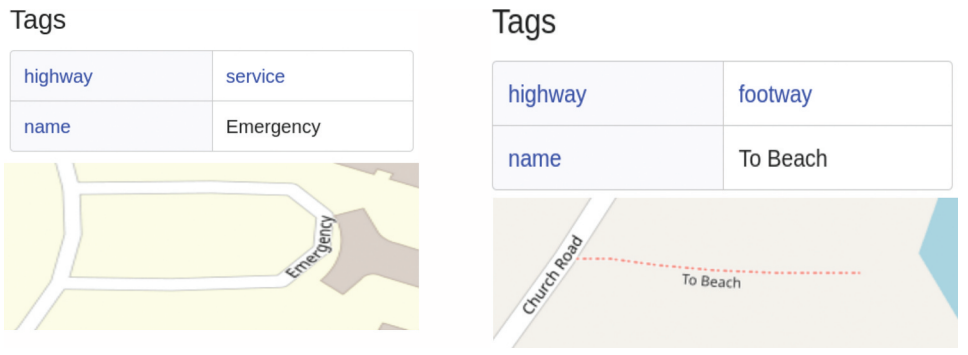

Figure 2. OpenStreetMap items with invalid names created by volunteers.

named by volunteers. They are a service way and a footway, respectively, which are incorrectly named by their purposes. That is, the service way is used for emergency and the footway is used to connect a beach. This is why they are not mapped or named in Google Map and Bing Map. The second is that the items named as a well-known single word (such as *'good'* and *'long'*), numbers, and English alphabet letters are removed. The well-known words are defined as the ones whose frequency is ranked among the top t^f in Word2Vec embedding vocabularies. Filtering is a common way used in gazetteer-based approaches to reduce false positives by using some heuristics, such as general words, stop words, and common person names (Middleton *et al*. 2018, Al-Olimat *et al*. 2018). This can lead to false negatives. However, we believe this is a reasonable trade-off made between false negatives and false positives. In the future, we will ease the constraints by introducing context features provided by a transformer model (e.g. BERT (Devlin *et al*. 2018)) in an unsupervised manner.

(2) **Augmentation**: For the place in the type of county, town, suburb, city, and state, the corresponding category words are removed to produce new positive samples if category words exist and the simplified name is not well known single words. For instance, for place name *'Jim Wells County'*, a new and valid place name is *'Jim Wells'*. The corresponding category name is added to the end of a place name if it is in the five types and has no category names. For instance, for place name *'Houston'* in the type of city, a new and valid place name is *'Houston city'*. Moreover, the items in the five types are given more attention by copying them C times since they are high-frequency and important places (e.g. *'Houston'*) and would be frequently used in microblogs. Furthermore, the roads at the highest level (e.g. country-level) marked as 'highway = motorway' and 'highway = trunk'[1] are also emphasized in the same way since they are the most significant roads in a country, which would be used and mentioned frequently.

(3) **Abbreviations**: In tweets, the usage of the abbreviation is quite common. For instance, *'Little Creek Road'* could be written as *'Little Creek rd'*. To deal with this issue, an English OpenStreetMap abbreviation dictionary[2] is used to extend the gazetteer by replacing the original word (e.g. city, county, highway, and bridge) with its abbreviation (e.g. cty, co, hwy, and bdge). The dictionary lists around 400 place-related common words and their abbreviations. Furthermore, we use

the abbreviation of places at the country and state level to extend the gazetteer, which is also provided on OpenStreetMap. For examples, the abbreviation of New York state is referred to NY and that of United States is referred to US.

(4) **Out-of-vocabulary**: In tweets, there can be expected some out-of-vocabulary words regarding the word embedding and gazetteers. We define that the combination of out-of-vocabulary words and category names can produce new place names. We randomly synthesize a word such as 'hiagnnamalnsw' to represent the out-of-vocabulary words. It is then combined with the category name to generate a new place name, such as *'hiagnnamalnsw street'*. This is to improve the detection rate of the place names containing out-of-vocabulary words.

3.1.2. Synthesizing negative examples

To train a place name classifier, we need not only positive examples (i.e. place names) but also negative ones. To generate negative examples, several rules are manually defined according to our prior knowledge about place names. Although these manually defined rules might be imperfect and can cause noisy negative examples, they would not affect the performance of the trained model. This is because deep learning algorithm can tolerate the modest amounts of noise in the training examples (Rolnick *et al*. 2017).

(1) **Sub set of place names**: The subset of a place name is regarded as a negative example if it is not included in the positive examples. For example, for the place name *'City of York'*, the subset *'City of'* and *'of York'* are treated as negative examples.

(2) **Reordering of place names**: Reorder the tokens of a place name to generate a negative example if it satisfies the following three conditions. 1) It is not included in the positive examples; 2) the position of at least three tokens changes; 3) the first and last words have changed, and the last word is not the location category name. For instance, for the place name *'National Park of New York'*, two negative examples are *'Park National New of York'* and *'Park New York of National'*.

(3) **Insert words before or after place names**: Insert a couple of general words at the beginning and end of a place name to generate new negative samples. For instance, for the place name *'National park of New York'*, the new negative examples could be *'it is National park of New York', "National park of New York is good'*, and *'c National park of New York 0000'*.

(4) **Combination of general words**: Randomly select several general words and combine them as negative examples if they are not included in the positive examples and the last word is not the location category name. For instance, *'i ft first'* and *'000 you are not b'* are negative examples.

(5) **Insert words before or after number-prefix-suffix words**: Numbers are first inserted before or after the number-prefix-suffix words to generate initial negative samples, named pre-suffix samples, such as *'0000 ft'* and *'am 0'*. Then, the general words and/or location category names are inserted at the beginning or end of the pre-suffix samples to produce new negative samples, such as *'at 0000 ft'* and *'Yes am 0 road'*. The pre-suffix samples and new negative samples are added to the training data if they were not included in the positive examples.

(6) **Combination of well-known words and location category names**: Choose a location category name and insert several well-known words at the beginning or end of the category name to generate a negative sample if it is not included in the positive examples. For instance, *'the city'*, *'by the way'*, *'university of a'*, and *'street is'* will be treated as negative examples.
(7) **Combination of English characters and numbers**: Randomly choose several English Alphabets and numbers and combine them as a negative sample if it is not included in the positive examples. For instance, *'t 000 p 0'* and *'0000 p 0'* will be treated as negative examples.
(8) **General words**: Each word in the general word list is treated as a negative example if it is not included in the positive examples, such as *'toilet'* and *'flood'*.
(9) **Insert general words between place names**: A couple of general words are inserted between two place names to generate new negative examples if they are not included in the positive examples, such as *'west little creek and florence'* from two place names *'west little creek'* and *'florence'* and one general word *'and'*.

3.2. Neural classifier

3.2.1. Classifier features

We use two kinds of features in the classifier. They are a general word embedding and six handcrafted features. The input layer concatenates all the features in a single vector. The generic word embedding is the pre-trained GloVe 50-Dimensional Word Vectors. We use six handcrafted features to represent the structure of an example explicitly, following (Guerini *et al*. 2018). They are (1) the actual position of the token within an example; (2) the length of the example; (3) the frequency of the token in the gazetteer ranging from 0 to 1; (4) the average length of the example containing a certain token; (5) the average position of the token in the example it appears in; and (6) the bigram probability regarding the previous token in the example.

The word length of a positive or negative example is forced to 20, and the one over 20 keeps only the last 20 words or below 20 is padded with 0 (word ID) at the beginning of the example. Therefore, the word vector dimension equals 56, while the dimension of the example matrix is 20×56.

3.2.2. Classifier module

The place names in the gazetteer can be divided into two types: (1) sequential place names (e.g. *'emergency operation center'* and *'formosan presbyterian church of greater houston'*) that consist of more than one word, follow structural characteristics, and contain category words (e.g. city, road, and center) and (2) non-sequential place names (e.g. *'Florence'*, *'New York'*, and *'Mississippi'*) that are short and do not have structural characteristics. For sequential place names, recurrent neural networks(RNN) are the better option since they specialize in sequential modeling (Zhou *et al*. 2015). CNN"-based models are better for non-sequential place names since they focus on local response features (Zhou *et al*. 2015). In this study, we apply the C-LSTM (Zhou *et al*. 2015) model for classifying place names, which combines a CNN and an LSTM to achieve the best of both. In C-LSTM, a CNN is first applied to extract higher-level representation. Then, a LSTM is applied to capture long-term dependencies over window feature sequences, respectively. The topology of the network is depicted in

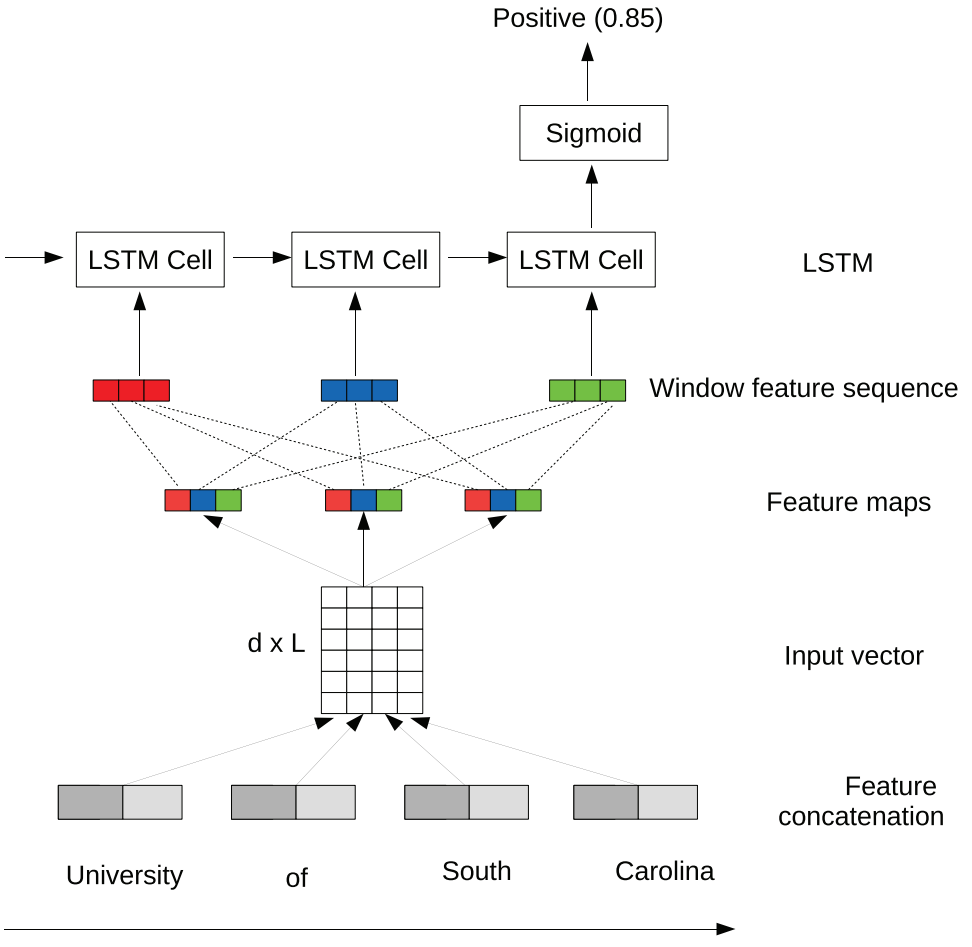

Figure 3. The architecture of the C-LSTM model for place classification. In the layer of feature concatenation, two types of features are concatenated, forming the next layer's input vector. *L* denotes the number of words in the input entity, while *d* denotes the dimension of the word vector. Blocks of the same color in the feature map layer and window feature sequence layer corresponds to features for the same window. The dashed lines connect the feature of a window with the source feature map. Dropout is then applied to the output of the last hidden unit of LSTM before the output layer (a sigmoid layer).

Figure 3. The input of the model is a sequence of words, such as *'University of South Carolina'* and *'I am'*. In the feature concatenation layer, two types of feature vectors for a word are concatenated to one vector. The dimension of the concatenated word vector is denoted by *d*. The length of the input sequence is forced to L, which is set to 20 in this paper. Therefore, the dimension of the input matrix is $d \times L$. Then, convolution is performed on the input matrix by using multiple filters. Thus, multiple feature maps are generated in the feature maps layer. The block of the same color represents the feature representation generated at the same window position of the input vector. Next, in the window feature sequence layer, the feature maps are rearranged by concatenating the feature representation for the same window. The generated new successive higher-order window representations are then fed

into LSTM. Next, we apply dropout to the output of the last cell of LSTM before the output layer (a sigmoid layer). The goal of the model is to classify the whole sequence as positive or negative. The output of the model is the confidence of the input sequence being a place name (positive probability).

3.3. Place name extraction from tweets

3.3.1. Tweet preprocessing

Given a tweet text, it is first tokenized. Then, the retweet handles, URLs, non-ASCII characters, and all user mentions in the tweet are removed since they do not contain place names. The numbers 0 to 9 are replaced with 0 and the token consisting of numbers are broken into numbers and words, such as to break *'hwy00'* to *'hwy'* and *'00'*. Misspelled tokens are then corrected by using the Symmetric Delete Spelling Correction algorithm (SymSpell).[3] The out-of-vocabulary words are replaced with the pre-defined word (such as 'hiagnnamalnsw'). Then, a statistical word segmentation algorithm (Matsuda *et al*. 2015) is used to break hashtags for location spotting as the number of locations in hashtags is significant. For instance, *#FlorenceFlood* is separated into two tokens *'Florence'* and *'Flood'*. Next, stop words including [ ] , % () ! ; : < > . are used to segment the tweet text into subsequences. For instance, *'Driving in from Sugar Land in the State of Texas! Any roads to avoid? '* are split into two subsequences *'Driving in from Sugar Land in the State of Texas'* and *'Any roads to avoid'*.

Each token in the subsequence is then assigned a POS tag by the CMU ARK Twitter tagger (Gimpel *et al*. 2010). The tagging scheme encompasses 25 tags, and each tag is denoted with a single ASCII character. Then, we rule that the POS tags of a valid place name consist of proper noun (∧), adjective (**A**), common noun (**N**), numeral (**$**), preposition (**P**), and other abbreviations and foreign words (**G**) but without the preposition (**P**) in the first and last position. When the set consists of only one word, its POS tag should not be pronoun (**O**) and adjective (**A**). We try to make this rule as general as possible, such that it covers all the place names and at the same time rules out as many invalid candidates as possible. Next, the valid subset of each subsequence, which follows the above POS rule, is selected. For example, the two sub-sequences are tagged as [*'Driving (**V**) in (**P**) from (**P**) Sugar (∧) Land (∧) in (**P**) the (**D**) State (**N**) of (**P**)* Texas (∧)] and [*Any (**D**) roads (**N**) to (**P**) avoid (**V**) '*]. **V** denotes verbs and **D** denotes determiners. According to our rule, *'Sugar'*, *'Land'*, *'Sugar Land'*, *'State'*, *'Texas'*, *'State of Texas'*, and *'roads'* are selected as valid subsets. Finally, the token of the valid subsets are converted to lower case.

3.3.2. Place name tagging

We used the trained classifier to calculate the confidence score of the valid subset of a subsequence (Line 9 in Algorithm 12). Then, we select as candidates the ones whose score is over a certain threshold denoted by s^t (Line 10 to 11 in Algorithm 1), which are then ranked in descending order according to the confidence score (Line 12 in Algorithm 1). Next, the ranking list is adjusted by moving the subset to the head of the one it includes if the score of the former is lower than that of the latter (From line 14 to 19 in Algorithm 1). That is, if candidate *a* (e.g. *'west florence'*) includes candidate *b* (e.g. *'florence'*), *a* is moved to the head of *b* if the score of *a* is lower than that of *b*. By doing this, the full place name can be extracted. Then, we select the top non-overlapping candidates as the recognized place entity (From line 20 to 22 in Algorithm 1). For example, given a subsequence *'Flood is serious*

at Little Creek West Florence' and its valid subsets selected by the POS rule, *little creek* (0.9) | *little creek west* (0.88) | *florence* (0.97)| *west florence* (0.95) will be selected as candidates if t^s is set to 0.8. The number following the candidate is the corresponding confidence score calculated by the classifier. The ranking result is thus *florence* (0.97) | *west florence* (0.95) | *little creek* (0.9) | *little creek west* (0.88). Then, the ranking is updated according to the inclusion relationship. The adjusted ranking is *west florence*(0.95) | *florence* (0.97) | *little creek west* (0.88) | *little creek* (0.9). Next, *west florence* is first selected, and *florence* and *little creek west* are then discarded because they overlap with the already selected one (*west florence*). Last, *little creek* is selected. Note that if the last token of the extracted place entity is in the word list ([area, region]), the last token is removed (From line 23 to 25 in Algorithm 1). They are indicators of a place name, but are unnecessary for a place name.

Algorithm 1: Place Name Tagging

Input:
M // trained classifier
S // token list of a sentence
s^t // score threshold
AREA // list of words that should be removed if in the end of an entity
Output:
Pla // extracted place names from *S*;

1: **procedure** TAGGER
2: $Pla \leftarrow null$;
3: $S_{sub} \leftarrow null$; // all the valid sub sets of *S*
4: $S_{can} \leftarrow null$; // candidates whose confidence score is over s^t
5: $P_{can} \leftarrow null$; // the confidence score of the candidates in S_{can}
6: $T \leftarrow null$; // the sorted result of candidates in S_{can}
7: generate all the valid sub sets of *S* and saved into S_{sub}
8: **for** $s \in S_{sub}$ **do**
9: $p \leftarrow M.calProbability(s)$ // calculate positive probability
10: **if** $p > s^t$ **then**
11: $S_{can} \leftarrow S_{can} \cup s$; $P_{can} \leftarrow P_{can} \cup p$
12: sort S_{can} according to P_{can} in descend order and saved to *T*
14: **while** $index < l$ **do**
15: $s \leftarrow T[index]$
16: search $T[0 : index - 1]$ to get first index (k) of child of *s*
17: **if** $k >= 0$ **then**
18: update *T* by moving $T[index]$ in front of $T[k]$
19: $index \leftarrow index + 1$
20: **for** $s \in T$ **do**
21: **if** *s* not overlap any element of *Pla* **then**
22: $Pla \leftarrow Pla \cup s$
23: **for** $l \in Pla$ **do**
24: **if** $l[-1] \in AREA$ **then**
25: delete $l[-1]$ and update *Pla*
26: **return** *Pla*

4. Experiments

In this section, we first detail the procedure of generating training data, the parameters used in the three main stages of the proposed approach, summarize test data, and introduce hardware configuration for running the proposed approach. Then, we compare and analyze the results achieved by our proposed approach and by existing approaches. Besides, the sensitivity analysis of several parameters such as the score threshold and filter size on the tagging accuracy is also performed. Finally, we compare the C-LSTM with the multi-channel CNN (Kim 2014) and Bi-LSTM (Guerini *et al*. 2018) based NER systems.

4.1. Training data preparation

We extract place names in the entire US and India from OSMNames.[4] OSMNames lists the place names derived from OpenStreetMap database, including both coarse and fine-grained places, such as country, state, city, town, village, suburb, neighborhood, mountain, street, creek, and bridge. Besides, the place names of public spaces and buildings, including libraries, airports, universities, hospitals, schools, churches, parks, and theaters across the US and India are also extracted by using the OpenStreetMap python tool[5] since they are important places but not sufficiently provided by OSMNames. This is intended to enable the detection of fine-grained places. In specific applications, the interested place level or type can be configured since the detected places can be linked to an item in gazetteers with geocoding, which has clear type or level information. For example, for traffic accident monitoring, the places at fine-grained levels specifically in the type of streets and roads can be configured. Furthermore, the abbreviation of the place at the country and state level is also provided on OpenStreetMap. We use the OpenStreetMap python tools to extract the abbreviations across the US and India, such as *'us'* for *'United Sates'* and *'tx'* for *'Texas'*. In total, 6.8 million initial positive examples are obtained from gazetteers, which are then augmented and extended to 37.6 million. Based on the augmented positive examples, negative examples are then synthesized. In total, 381.3 million negative examples are generated.

We set several parameters for the proposed approach (shown in Table 1), with which the best performance can be achieved. Moreover, we conduct experiments to analyze the impact of some parameters (i.e. filter size and score threshold) on the performance of the proposed approach. During the procedure of generating training examples, the count threshold for the location category name (t^l) is set to 400, the frequency threshold for the well-known words (t^f) is set to 26000, and the increased copies for significant places (C) is set to 150. As for C-LSTM model, a single convolution layer is used with the number of filters (f^n), the filter length in CNN (f^l), the memory dimension in LSTM (d^l), and the dropout rate (dr) are set to 120, 1, 120, and 0.5, respectively. During the model test procedure, the score threshold (t^s) for choosing the valid candidate is set to 0.78.

Note that to deal with the imbalanced data issue, a weighted loss computation strategy is utilized. Specifically, a heavier penalty would be placed on the misclassification of the minority class (positive samples), by assigning the minority class a larger weight. The weight equals the ratio of the number of negative and positive samples.

Table 1. Parameters of the proposed approach.

Procedure	Param	Value
Training data	Count threshold for location	400
Generation	Category name (t^l)	26000
	Frequency threshold for Well known words (t^f)	
	Increased copies for Significant places (C)	150
Model training	Number of filters (f^n)	120
	Filter length in CNN (f^l)	1
	Memory dimension In LSTM (d^l)	120
	Dropout rate (dr)	0.5
	Update function	SGD
	Learning rate	0.001
	Number of epochs	12
	Batch size	1024
Model test	Score threshold for Choosing valid candidate (t^s)	0.78

4.2. Test data

To demonstrate the effectiveness of the proposed place name extractor, we used three publicly available tweet datasets with annotated place names from Al-Olimat *et al*. (2018) corresponding to the 2016 Louisiana flood, the 2016 Houston flood, and the 2015 Chennai flood, respectively.[6] The summary of the test data is shown in Table 2. We found a few missing location mentions in the annotated data due to human errors. Although the proposed model can detect most of these missing locations, such as *'Tchefuncte River'*, *'I-10 East'*, and *'LDS Church'*, we still consider the detected location mentions as false positives.

In the dataset, location mentions were divided into three types: *inLOC, outLOC*, and *ambLOC*, representing the locations inside the area of interest (e.g. 'Baton Rouge'), outside the area (e.g. 'New York') in the context of Louisiana flood, and ambiguous locations (e.g. 'my house'). Similar to the evaluation approach in Al-Olimat *et al*. (2018), we only evaluated the systems on the inLOC and outLOC location mentions. We adopted the standard comparison metrics: Precision, Recall, and F1-Score. In the case of overlapping or partial matches, we penalize the approaches by adding 1/2 FP (False Positive) and 1/2 FN (False Negative) (e.g. if the tool marks 'The Houston' instead of 'Houston').

Apart from the three annotated datasets, we also apply the proposed approach to one un-annotated dataset, which corresponds to the 2018 Florence Hurricane. It contains over 100,000 tweets. According to our observation, our proposed approach still works well on the dataset. The tagging result has been published together with the code.

Table 2. Summary of test datasets.

	Louisiana	Houston	Chennai	Total
Number of tweets	1500	1500	1500	4500
Number of place names	2918	4177	4589	11684
Proportion of inLoc	66%	66%	75%	70%
Proportion of outLoc	13%	7%	4%	7%
Proportion of ambLoc	22%	27%	21%	23%

4.3. Hardware

In our institute, there are one CPU cluster and one GPU cluster. The CPU cluster includes 116 nodes, and each node contains 2 Intel Xeon platinum 8260 with 48 cores. The GPU cluster includes 4 nodes, and each node contains 8 NVIDIA Tesla V100 GPUs. Our approach consists of three main steps. We use one CPU core to execute the training data generation step, which takes 4.4 hours. The maximum memory consumption is 10 GB. We use PyTorch as a training framework and use one GPU core to train the model based on the positive and negative examples, which takes 3.5 hours for one epoch. The maximum memory consumption is 80 GB. We use one CPU core to load the model and identify the place names from tweet texts, which takes around 8 minutes in total for the three data sets.

4.4. Tagging result

We compare our approach with competitive place name extraction systems, including Google NLP, Stanza,[7] CLIFF,[8] NeuoTPR (Wang *et al*. 2020), CNN-based approach (Kumar and Singh 2019), and LNEx (Al-Olimat *et al*. 2018) on the same testing dataset. Google NLP and Stanza are general NLP tools while CLIFF is a NER tool. The other tools are developed specifically for geo-tagging or place name extraction. Note that LNEx was compared with many location extractors, such as DBpedia Spotlight (Mendes *et al*. 2011), Yahoo PlaceFinder,[9] Geo-locator 3.0 (Gelernter and Balaji 2013), and Geoparsepy (Middleton *et al*. 2013) on the same datasets. The results showed that LNEx achieved the best. Thus, in this paper, we avoid adding the comparison that the LNEx paper already provided. Instead, we just compare with LNEx and another set on the same dataset. We considered the entities of type Location, Organization, and Address as location mentions from Google NLP. We used the NER tool in Stanza and kept the entities of type Location, Organization, Facility, and Geopolitical Entity. As for CLIFF, we kept organization and place mentions and ignored the focus and people categories (since we are interested in location mentions only and are not interested in the tweet's focus). The above three tools are publicly available. Thus, we directly utilized their provided API to extract place names. For NeuroTPR we simply used their trained model and implementation to tag the dataset.[10] The CNN-based approach was re-implemented and evaluated in two ways. We used the parameters suggested by Kumar and Singh (2019) and also changed them to obtain the best F1 score. The first evaluation method follows Kumar and Singh (2019), specifically 10-fold cross-validation based on the three datasets, named 10-fold CNN. Each dataset was divided into ten groups. In each test, one group is used as test data and the remaining nine groups are used as training data. The average result of the ten tests is used as the evaluation result. The second evaluation method is to use two datasets (e.g. Houston and Louisiana) as the training data and the third one (e.g. Chennai) as the test data, named transfer CNN. This is to evaluate the transferability of the trained model.

Table 3 contains the full results of different approaches. On all of the three datasets, our proposed approach achieves the best F1-score of 0.82, 0.84, and 0.86, respectively. The second-best performing approach was LNEx, achieving the F1 scores of 0.82, 0.76, and 0.85, respectively. Note that, for the test data in a certain region (e.g. Houston), LNEx only used the OpenStreetMap data in this narrow region, which can dramatically reduce

Table 3. Tagging result of multiple place name extractors.

		Louisiana			Houston			Chennai			Avg
		P	R	F	P	R	F	P	R	F	F
General	Google NLP	0.36	0.72	0.48	0.38	0.61	0.47	0.43	0.60	0.50	0.48
	Stanza	0.43	0.63	0.51	0.59	0.35	0.44	0.54	0.36	0.43	0.46
Tools	CLIFF	0.82	0.79	0.80	0.80	0.49	0.61	0.74	0.37	0.49	0.63
Location	NeuroTPR-1	0.83	0.48	0.61	0.77	0.28	0.42	0.81	0.32	0.46	0.50
Extractors	NeuroTPR-2	0.55	0.60	0.57	0.65	0.36	0.47	0.70	0.39	0.50	0.51
	10-fold CNN	**0.93**	0.53	0.68	**0.87**	0.64	0.74	0.86	0.59	0.70	0.71
	Transfer CNN	0.78	0.33	0.47	**0.87**	0.40	0.55	0.66	0.08	0.14	0.39
	LNEx	0.83	**0.81**	**0.82**	**0.87**	0.67	0.76	**0.91**	0.80	0.85	0.81
	GazPNE	0.88	0.77	**0.82**	**0.87**	**0.81**	**0.84**	0.87	**0.85**	**0.86**	**0.84**

the interference of ambiguous place names. Conversely, our model is trained on OpenStreetMap data in the entire US and India, which includes much more ambiguous place names, such as Donald and Hillary. This dramatically increases the false positives. When we narrow the region by using the OpenStreetMap data in the south part of the US and India, which is still much larger than that of LNEx, our model can achieve an average F1 score of 0.87. However, to fairly compare our model with the other approaches, we use the OpenStreetMap data from the entire US and India. Google's and Stanza's general-purpose NER systems underperformed the other systems even though we did not penalize them for missing the hashtags' location mentions. NeuroTPR-1 came right after them, achieving, on average, an F1 of 0.50 (NeuroTPR-2 achieved an F1 of 0.51 when not counting location mentions in hashtags as false negatives). The approach still suffers from the sparsity of labels, although the model was trained on a large set of labeled locations from Wikipedia. The 10-fold CNN achieves an acceptable tagging result. However, it requires that test data sets should be similar to training data sets, such as the shared place names appearing in both test and training data sets. It is impractical in a real application since it relies too much on chosen training datasets and has poor transferability. This can be proved by the low tagging performance achieved by the transfer CNN. Chennai dataset is different from the Louisiana and Houston datasets. Therefore, when we use Houston and Louisiana datasets to train a model, nearly no place names can be recognized from Chennai dataset by the model. The results suggest that the above two statistical learning-based approaches are struggling with generalizing across different datasets and that they need much more data to train a stable model.

Compared to LNEx, our method achieved higher precision and recall across the board (except for the recall on Louisiana dataset and precision on the Chennai dataset). That suggests that the LNEx method of skip-gram-based gazetteer augmentation was not good enough to improve recall and was not that accurate (generating noise), ultimately harming precision. Conversely, our proposed approach is more powerful and robust because of two reasons. The first is that it understands the characteristics of multi-word place names such that many multi-word place names not included in the gazetteer can be recognized, such as *'Chennai Airport'*, *'Keith Weiss Park'*, *'Katy Hockley cutoff'*, *'Hidden Valley Church of Christ'*, *'Lake 610'*, and *'Paint Creek Bridge'*. The second is that the simple POS rule can deal with some hard examples for the model and thus further improve the robustness of the proposed approach. For example, *'me'* is the abbreviation of Maine state and included in the positive examples. Our model would always classify it as positive. However, 'me' could also refer to persons when it is pronoun (O). In this case, the defined

POS rule can judge if it refers to persons or places. Another example is 'us', which can refer to persons or places based on its POS tag (pronoun or proper noun). The two aspects mainly contributed to the higher F1-score of the proposed approach compared to LNEx and the other systems.

75% of the place names in the dataset are unigrams, the remaining 14%, 6%, 3%, and 2% are bigrams, 3-grams, 4-grams, and more than four, respectively. The detection rate of the annotated place names with different lengths is illustrated in Figure 4. The detection rate of the one-word place name is the highest on all three datasets. The other types of place names' detection rates are all over 0.5, suggesting an acceptable performance of detecting multi-word place names.

Furthermore, we exhaustively analyzed incorrect detection by GazPNE. A selection of 17 representative tweets along with tagging results is shown in Table 4. Generally, incorrect detection has the following six types. (1) Many person names are detected as place names, such as 'Katy' and 'Nelson' in the 5th tweet. Our model does not utilize further context information. Thus, all the ambiguous names (e.g. 'Clinton', 'Donald', and 'Washington') will be judged as a place since they appear in gazetteers. This issue can be mitigated by removing full person names from a text with a person name dictionary as Middleton *et al*. (2018) did. (2) Some one-word place names which are not included in the gazetteer cannot be recognized, such as 'Binz' in the 2nd tweet, 'khou11' in the 6th tweet, and 'hou' in the 7th tweet. We found 291 such place names, which are unseen abbreviations or jargons (e.g.'hou', 'nws', and 'kou11') in the Houston dataset, while there are 201 'hou' which is the abbreviation of 'houston'. They are not presented in gazetteers. The correct detection of them can increase the F1-score of the Houston dataset to 0.90. (3) The long place name that does not own lexical and structural characteristics cannot be detected if they are not included in the gazetteer, such as 'pondy bazaar' in the 13th tweet. (4) The erroneous break of hashtags by the statistical word segmentation algorithm (Matsuda *et al*. 2015). The used algorithm breaks '#lawx' into 'law' and 'x' (in 3-th tweet) since this combination is more probable, but the correct break should be 'la' and 'wx'. The proposed algorithm can detect 'la', but the incorrect

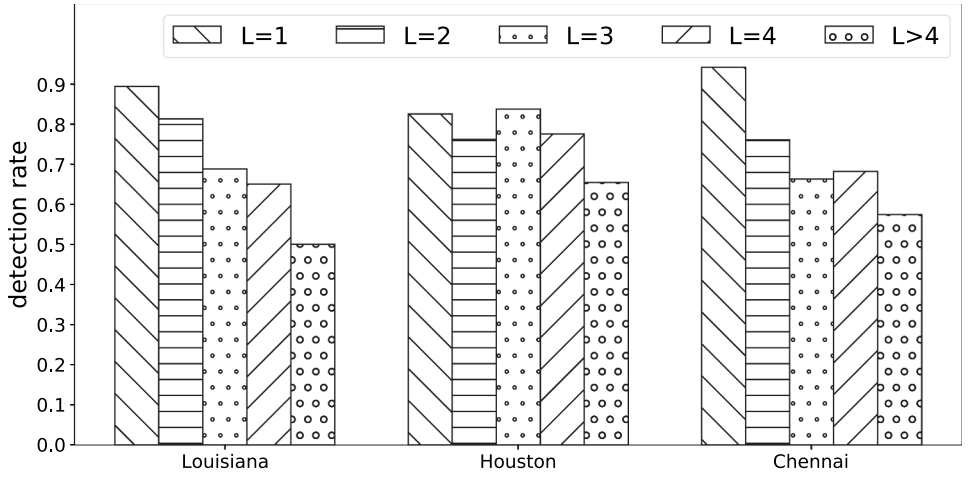

Figure 4. Detection rate of the place names with variant length on three datasets.

Table 4. Example of tweets and tagging results of the proposed approach. Bond texts refer to true place names in tweets. Underline texts refer to incorrectly detected place names.

Tweet	Extracted place name
RT @AaronKPRC: #TRAFFIC ALERT: **I-10** closed in both directions at **Taylor St**. High water across the interstate. #**houston**flood @KPRC2	(I-10), (Taylor St), (houston)
RT @joysewing: **Highway 288** at **Binz** still under water. #**houston**flood https://t.co/7AcMebF3yX	(Highway 288), (houston)
HT @NWSNewOrleans: Parts of **I-55** CLOSED near **Amite City**, **LA** due to #flooding. #**la**wx https://t.co/QQl8Xl8OdX	(Amite City), (I-55),(la)
Mississippi medical center receives record obesity grant: The **University of Mississippi** Medical . . . https://t.co/lBsoJ8iHVc #**mississipp**	(Mississippi medical center), (University of Mississippi), (mississipp)
RT @drkatynelson: Advice from Dr. Katy J. Nelson on how to help the people and animals Affected by South **Louisiana**'s flooding	(Louisiana), (Katy), (Nelson)
Per **emergency operations center**: city working On joint effort to relocate affected **greenspoint** Residents #**khou11** #**houston**flood	(emergency operations center) (greenspoint), (houston)
RT @wxJonDJ: Flooded **Cypress Creek** near **Stuebner Airline Road** in **Spring, TX** #**houston** Flood #**hou**wx https://t.co/b69EbnaWmN	(Cypress Creek), (Stuebner Airline Road),(TX), (houston)
FWD cancels Flood Warning for North **Bosque River** at **Valley Mills** [**TX**] https://t.co/aB4DwLzh5X #**tx**wx #ntxwx	(North Bosque River), (Valley Mills), (TX), (tx)
Anyone have any pictures of what the **Addicks reservoir** at **highway 6** and also **eldridge** looks Like right now? #**houston**flood #**houston**weather	(addicks reservoir), (highway 6), (eldridge), (houston), (houston)
#CircleNews We will be working with The **Village Life Center** in #**Louisiana** to help bring Awareness to the . . . https://t.co/uKqjt1tFfM	(Village Life Center), (Louisiana)
#**BuffaloBayou** is overflowing and flooded **Memorial Drive**. If you zoom in, there's a car. #**houston**flood #flood2016 https://t.co/V0mDTCobz8	(Buffalo Bayou), (Memorial Drive), (houston)
Radar Storm Total vs. **Harris County Flood Control District** Rain Gauges @hcfcd. Radar overall Underestimated. #**hou**wx https://t.co/4TNk0hSsEa	(Harris County), (Flood Control District)
RT @AkshayK88: need immediate rescue New no 32 (old no) 27 **Raman street, t.nagar, Chennai** 17. Behind **Holy Angels convent, pondy bazaar** -two	(Raman street), (Chennai), (Holy Angels convent), (t.nagar)
#**Chennai**Rescue-my cousins also stuck at **Egret park, anand nagar** extn, **thoraipakkamm**. Ground Floor under watr.Dey r now at 1st floor.6 people.	(Chennai), (egret park), (anand nagar extn)
RT @PIB_India: #**Chennai**Floods: Railway Shuttle service from **Chennai beach station** to **Arakkunnam station** in every 45 minute.	(Chennai), (Chennai beach station), (Arakkunnam station)
RT @Hannahsmiles1D: #**houston**flood Scary experience Seeing someone trapped near **290**. Such a shame of All this damage. I almost got trapped b.	(houston)
METRO has deployed buses to bring people in need Of shelter to **MO Campbell Center**. #SD6 #**hou**wx	(MO Campbell Center)

break causes the missing detection. 67 '#lawx' have been found in the Louisiana dataset and the correction of this error can increase the F1-score for the Louisiana dataset from 0.82 to 0.84. (5) Some one-word place names (such as 'Spring', '269') have been filtered from the gazetteer because they are well-known general words. This has led to the missing detection of 'Spring' in the 7th tweet and '290' in the 16th tweet. (6). Some multi-word place names are detected as multiple sub-place names, such as 'Harris

County Flood Control District' in the 12th tweet, which was detected as 'Harris County' and 'Flood Control District'. The first, second, and fifth issues contribute to the most detection errors, which are also the main drawback of our proposed approach due to the lack of context information.

Furthermore, we analyzed the contribution of several tricky strategies adopted in our approach, which can improve the performance to some degree.

Out-of-Vocabularies: This strategy is applied in the positive example augmentation procedure by generating positive examples and in the tweet prepossessing procedure. The result shows this has led to 0.1% and 0.2% increase of the F1-score for the Houston and Chennai datasets, respectively, by detecting the place names containing out-of-vocabulary words, such as *'Sriramachandra Medical College'*, where *'Sriramachandra'* is an Out-of-Vocabulary word and *'Chembarakam Lake'*, where *'Chembarakam'* is an Out-of-Vocabulary word.

Removing prefix or suffix: This is applied in the online tagging stage by removing the meaningless words at the end of place names for toponym resolution. The result shows this has led to 0.3%, 0.9%, and 0.1% increase of the F1 score for the Louisiana, Houston, and Chennai datasets, respectively. For example, *'area'* is removed from *'Meyerland area'*, which is a detected place name by the proposed approach. *'Meyerland area'* and *'Meyerland '* are different from the perspective of information retrieval. However, they refer to the same place in location extraction, i.e. a neighborhood in Houston, which will be geocoded as a boundary box corresponding to 'Meyerland' in toponym resolution.

4.5. Sensitivity analysis

4.5.1. Impact of score threshold on tagging accuracy

The score threshold (t^s) is a key parameter at the place name tagging stage. Thus, the impact of t^s on the tagging accuracy is analyzed. In the experiment, the three datasets adopt the same t^s value, which varies from 0.5 to 0.95 at an interval of 0.02. Figure 5 shows the result achieved by the proposed approach as the change of t^s. Typically, a lower t^s value would lead to higher false positives, while a higher t^s value would lead to lower true positives. We can see when t^s is set in the range of [0.7,0.88], the highest F1-score is achieved. The lowest F1-score is achieved when t^s is set to 0.95. The results reveal that t^s would affect the tagging result, but on the whole, the performance remains stable for a broad range of t^s values.

4.5.2. Impact of filter length on performance of C-LSTM

The impact of filter length (f^l) on the tagging accuracy is investigated. The filter length of 1, 2, 3, and 4 are adopted with only one convolutional layer and the same filter length. The training epoch is set to 12. During the training procedure, 16 million examples are used as the test set. The test accuracy of different filter length configurations is shown in Table 5. We can see the filter length of 1 is the best configuration for the test accuracy (at 0.9972), which is higher than that of the other configurations. However, due to the existence of noise data in the training examples caused by inaccurate rules, the test accuracy cannot fully reflect the trained model's performance, especially when the test accuracy of these configurations is close. Therefore, they are further evaluated and compared by applying trained models in extracting place names from the three test datasets. In the experiment,

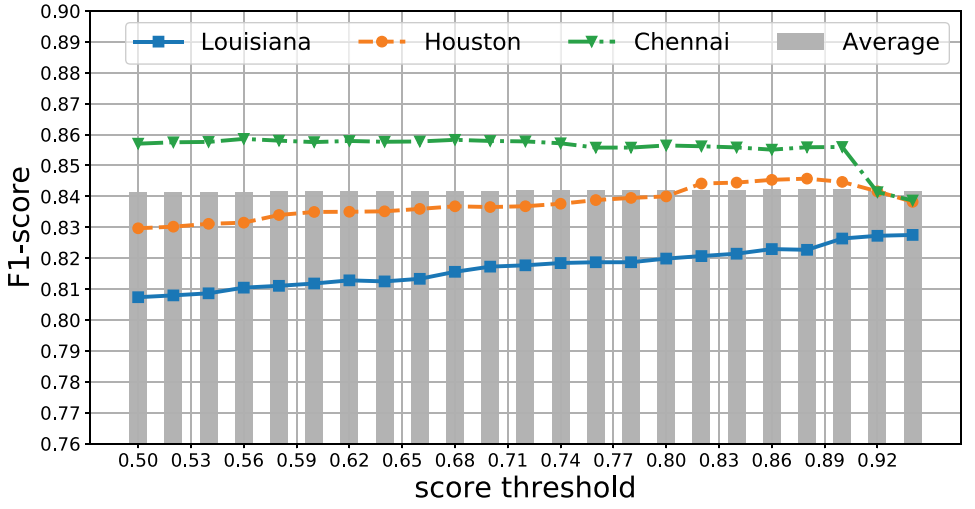

Figure 5. Impact of t^s on the F1-score for three datasets.

Table 5. Impact of filter length on model performance.

filter length	Test accuracy	Average F1 score	Best t^s
$f^l = 1$	0.9972	0.84	0.78
$f^l = 2$	0.9968	0.83	0.72
$f^l = 3$	0.9964	0.83	0.74
$f^l = 4$	0.9961	0.83	0.78

t^s varies from 0.5 to 0.95 at an interval of 0.02. The highest F1 score achieved by the four configurations is shown in the third column of Table 5 with the best t^s value in the fourth column. Still, filter length of 1 performs the best. This is because most of the place names in gazetteers are short, as shown in Figure 6. We can see the length of most of the place names lies in the range of [1,4]. Therefore, a small filter length can achieve better performance.

4.5.3. Comparison of C-LSTM, CNN, and Bi-LSTM

In this study, we adopt the C-LSTM model, which combines CNN and LSTM. An extra experiment is conducted by training a classifier using the classic multi-channel CNN (Kim 2014) and one layer bidirectional-LSTM models (Guerini *et al.* 2018) for place entity recognition with the same training and test data. The key parameters for the CNN-based model include the number of channels, the filter length in each channel, the number of filters, and the dropout rate, which are set to 3, [1,2,3], 120, and 0.5, respectively. The key parameters for Bi-LSTM include the memory dimension and the dropout rate, which are set to 120 and 0.5, respectively. The training epoch is set to 12. 16 million examples are used as the test set. The achieved test accuracy of the three models is shown in Table 6. We can see C-LSTM and CNN outperform Bi-LSTM in test accuracy. This is mainly because most of the place names are short, such that lexical characteristics dominate the structural characteristics of place names. Moreover, six structural features are manually defined, which can represent the structural characteristics of place names to a certain degree. This is why CNN performs

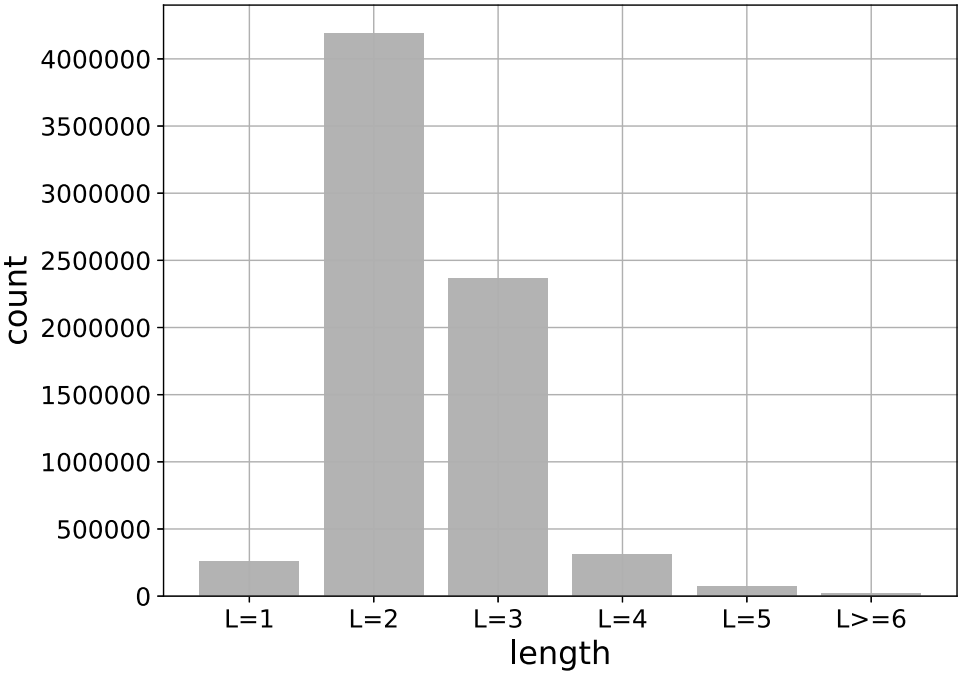

Figure 6. Distribution of length of place names in gazetteers.

Table 6. Comparison of different models.

model	test accuracy	average F1 score	best t^s
C-LSTM	0.9972	0.84	0.78
CNN	0.9965	0.82	0.78
Bi-LSTM	0.9572	0.80	0.86

better than Bi-LSTM. Furthermore, the trained models are used to extract place names from the three tweet datasets with varied t^s from 0.5 to 0.95 at an interval of 0.02. The best F1-score achieved by the three models is shown in the third column of Table 6 with the best t^s in the fourth column. Still, C-LSTM performs the best, while Bi-LSTM performs the worst.

5. Discussions

In this section, we further discuss some concerns, issues, and challenges of our study, including the impact of word-embedding configurations on model performance, inaccurate rules, and the challenge of geocoding.

Word-embedding: Some studies proposed specific word embeddings for geospatial data hierarchical structures representation and semantic similarity calculation (Gao and Yan 2018, Dassereto *et al*. 2019). They were proved to be more powerful than general word-embedding such as Glove in spatial similarity measurement. However, our task is different, which does not need the semantic similarities of place names but focuses on distinguishing place names from non-place names. Thus, we just use general word-embeddings (i.e. Glove). However, these studies inspired us to analyze the impact of different word-

embedding configurations on model performance. Thus, an extra experiment is conducted. In the experiment, we combine one general embedding (i.e. Glove, Google, or BERT) with optional domain-specific embedding (Guerini *et al*. 2018) learned from gazetteers (OpenStreetMap) by using (Mikolov *et al*. 2013a), named OSMEmb. Six configurations are compared, and they are Glove+OSMEmb, Google+OSMEmb, BERT+OSMEmb, Glove, Google, and BERT. The results show these different configurations achieve the same performance. The reason is that the purpose of word embeddings is to make sure that similar words (such as road and street) have similar representations such that even if some words or segments are not included in training data, the model can still generalize to the test data which includes the unseen words or segments. That is, word embedding is useful when only limited training data is available. However, in our case, we have abundant training examples (nearly 7 million positive examples and 380 million negative examples). The training data contain most of the place-related words. Thus, with one general word representation, the model can already distinguish place names from non-place names.

Inaccurate rules and noisy data: In this study, negative training examples are generated by a couple of heuristics or explicit rules, which are noisy. For example, according to our rule, negative example *'christian college of kansas'* is generated by inserting *'christian'* before *'college of kansas'*. No spatial item in the real world is named *'christian college of kansas'*. However, it is still a valid place name since it follows the norms of place names. Deep learning algorithms can learn the characteristics of place names based on the majority of correct training examples. Thus, many unseen place names can be classified as positive by the model, such as *'Hidden Valley Church of Christ'*. However, it is still unclear if the noisy data has harmed the performance of the model, whether the deep learning model is tolerant of imperfect rules, or whether the trained deep learning model can improve the rules or not. These issues are significant since noisy data, sparse training data, and weak interpretability are the biggest challenge faced in deep learning algorithms. The fusion of rules and deep learning in this study has inspired us to investigate these issues in the future.

Location extraction: Location extraction consists of two steps: toponym recognition and toponym resolution. This study focuses on toponym recognition, leaving the second step for the future, which is more challenging (Gritta *et al*. 2018). In toponym resolution, gazetteers or map API (such as Google Map API) will be searched to find the place name's coordinates. However, this issue is not solved well due to the existence of Geo-ambiguities and place name variants. Geo-ambiguities refer to the situation that a place name might have multiple matches of geographic locations, such as Manchester, NH USA versus Manchester, UK. To overcome this challenge, the cues about the correct match of the place can be utilized such as the administration level (e.g. country, state, city, suburb, town, and county) and population size of the place since they reflect the importance or popularity of the place. Important places are frequently used in social media sites and should be given a higher confidence score. The places in the same tweet text or in the tweet text of the same semantic cluster are also strong features of the correct match (Karimzadeh *et al*. 2019). Place names have variants such that detected place names may not have exact matches in gazetteers. To solve this issue, we plan to calculate the semantic distance between a detected place name and the geo-items in gazetteers and infer its geolocation based on the semantic nearest neighbors. The idea is inspired by (Di Rocco *et al*. 2021), which geolocates a posted message at sub-city levels. It extracts the semantic of toponyms in the post from a geographic gazetteer and then embeds them

into a metric space, which captures the semantic distance among them. The semantically closest toponyms to a message are then identified and clustered with respect to their spatial locations. Moreover, less attention has been paid to the comparison of toponym resolution mainly because benchmark data is lacking. Fortunately, Wallgrün *et al*. (2018) has collected and published a corpus of geo-annotated tweets, which enables the evaluation of toponym resolution approaches.

6. Conclusion

Current approaches for extracting place names from microblogs face crucial problems: rule-based methods do not generalize, gazetteer-based methods cannot detect unseen multi-word place names, and machine learning methods lack sufficient data, which is costly to annotate on scale. To overcome these issues, we propose a robust approach, which combines gazetteers, rules, and deep learning to achieve the best of all. It can learn the intrinsic features of multi-word place names from gazetteers without the need for any manually annotated data. The approach was evaluated on three public tweet datasets with 9,026 place names in total. An average F1-score of 0.84 is achieved. The approach has the potential of being generalized to regions worldwide because OpenStreetMap is world-widely available. The study introduces a new way of combining rules with deep learning, which produces abundant training examples by rules. This can be applied to other domains. One drawback of the proposed approach is that it cannot deal with ambiguity issues since we do not use any context features. For instance, *'Washington'* could be a place or person name, depending on its contexts in texts. In the future, we plan to solve this issue by fusing the intrinsic features of an entity provided by our model and the context features of the entity in a text provided by a transformer model (e.g. BERT) in an unsupervised manner.

7. Data and codes availability statement

The data and codes that support the findings of this study are available in github with the identifier at the link https://github.com/uhuohuy/GazPNEhttps://github.com/uhuohuy/GazPNE

Notes

1. highway https://wiki.openstreetmap.org/wiki/Key:highway
2. OpenStreetMap placewiki.openstreetmap.org/wiki/Name_finder:Abbreviations
3. SymSpell https://github.com/wolfgarbe/symspell
4. OSMNames https://osmnames.org/
5. OpenStreetMap_python_tools https://github.com/mocnik-science/osm-python-tools
6. The data can be obtained from https://rebrand.ly/LocationsDataset
7. https://stanfordnlp.github.io/stanza/
8. https://cliff.mediacloud.org/
9. www.programmableweb.com/api/yahoo-placefinder
10. We did not retrain the model, we simply used the model provided by the authors at https://github.com/geoai-lab/NeuroTPR

Disclosure statement

No potential conflict of interest was reported by the author(s).

Notes on contributors

Xuke Hu received his PhD degree in Geoinformatics from Heidelberg University, Germany in 2020. He is now a postdoctoral researcher at the Data Science Institute of the German Aerospace Center (DLR). His research interests include indoor localization/navigation/mapping, VGI, social sensing, earth observation, and disaster management.

Hussein Al-Olimat is a senior NLP Data Scientist at Tempus Labs. He obtained his PhD. in Computer Science from Wright State University in 2019 and was part of the Kno.e.sis research center working on multi-disciplinary projects in the areas of healthcare and decision support. In his research, he pursued multifaceted syntactic, semantic, and pragmatic information extraction techniques. His vision paper on geocoding spatial expressions was recognized by the Computing Community Consortium (CCC) as a Blue Sky vision idea, in which he tries to tie NLP to GIS.

Jens Kersten received his doctoral degree (Dr.-Ing.) in remote sensing and computer vision and his diploma in geodesy from the Technical University of Berlin, Germany. His research at the German Aerospace Center (DLR, 2008-2013) and the Bauhaus-Universität Weimar (2014-2018), Germany, was related to adjustment calculus, information extraction from optical and SAR imagery, emergency and crisis mapping, traffic monitoring, and photogrammetric computer vision. In 2018, he joined DLR's Institute of Data Science and currently researches new methods on web- and social media data analysis.

Matti Wiegmann is a PhD researcher at the chair of Web technology and Information Systems (Webis) of the Bauhaus-Universität Weimar. His research interests include language processing and information retrieval systems for social media data, particularly information collection, extraction, and validation for disasters and other hazardous incidents.

Friederike Klan is heading the Citizen Science Dept. at the DLR Institute of Data Science. She has a scientific background in computer science with a specialization on knowledge and data management. The focus of her work is on software co-creation, mobile data collection, data interoperability and data quality as well as data privacy aspects in Citizen Science projects. In her academic career, she has initiated and managed 12 scientific projects and published +40 scientific papers. She is initiator and active member of several national and international scientific working groups and has (co-)organized +25 scientific workshops and conference sessions on data management and Citizen Science with a focus on data-centric and legal aspects.

Yeran Sun is an Assistant Professor of Swansea University. His research fields include urban informatics, urban remote sensing, urban transport, health geography, and big data analytics. He is particularly interested in the combination of conventional and emerging forms of geospatial data in support of developing sustainable and healthy cities.

Hongchao Fan is professor for 3D Geoinformatics at the Department of Civil and Environmental Engineering at the Norwegian University of Science and Technology (NTNU). He received his master's degree in Geodesy and Geoinformatics at the University of Stuttgart and obtained his PhD at the Technical University of Munich in Germany. After that, he worked as Group Leader for 3D Data Infrastructure at the Heidelberg University for six years. In 2018, he started his work as professor at NTNU in Trondheim, Norway. His research interests include 3D city modelling, spatial data mining from VGI data and laser scanning.

ORCID

Hongchao Fan http://orcid.org/0000-0002-0051-7451

References

Alexander, D.E., 2014. Social media in disaster risk reduction and crisis management. *Science and Engineering Ethics*, 20 (3), 717–733. doi:10.1007/s11948-013-9502-z

Al-Olimat, H., *et al*. 2018. Location name extraction from targeted text streams using gazetteer-based statistical language models. *In*: *Proceedings of the 27th International Conference on Computational Linguistics*, Santa Fe, New Mexico, 986–1997. https://www.aclweb.org/anthology/C18-1169 .

Beall, J., 2010. Geographical research and the problem of variant place names in digitized books and other full-text resources. *Library Collections, Acquisitions, & Technical Services*, 34 (2–3), 74–82. doi:10.1080/14649055.2010.10766263

Bontcheva, K., *et al*. 2013. Twitie: an open-source information extraction pipeline for microblog text. *In*: *Proceedings of the International Conference Recent Advances in Natural Language Processing RANLP 2013*, Hissar, Bulgaria, 83–90.

Cheng, Z., Caverlee, J., and Lee, K., 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*, Toronto, Canada, 759–768.

Das, R.D. and Purves, R.S., 2019. Exploring the potential of Twitter to understand traffic events and their locations in Greater Mumbai, India. *IEEE Transactions on Intelligent Transportation Systems*, 20 (7), 2566–2583. doi:10.1109/TITS.2018.2868182

Dassereto, F., *et al*. 2019. Evaluating the effectiveness of embeddings in representing the structure of geospatial ontologies. *In*: *International Conference on Geographic Information Science*, Limassol, Cyprus, 41–57. Springer.

Devlin, J., *et al*., 2018. Bert: pre- training of deep bidirectional transformers for language understanding. *arXiv Preprint arXiv:1810.04805*.

Di Rocco, L., *et al*., 2021. Sherloc: a knowledge-driven algorithm for geolocating microblog messages at sub-city level. *International Journal of Geographical Information Science*, 35 (1), 84–115. doi:10.1080/13658816.2020.1764003

Dutt, R., *et al*. 2018. SAVITR: a system for real-time location extraction from microblogs during emergencies. In *Companion Proceedings of the The Web Conference 2018*, Lyon , France, 1643–1649.

Finkel, J.R., Grenager, T., and Manning, C., 2005. Incorporating non- local information into information extraction systems by gibbs sampling. *In*: *Proceedings of the 43rd annual meeting on association for computational linguistics*, 363–370. Ann Arbor, Michigan: Association for Computational Linguistics.

Gao, S. and Yan, B., 2018. Place2vec: visualizing and reasoning about place type similarity and relatedness by learning context embeddings. *In*: *Adjunct Proceedings of the 14th International Conference on Location Based Services*, 225–226. Zurich, Switzerland: ETH Zurich.

Gelernter, J. and Balaji, S., 2013. An algorithm for local geoparsing of microtext. *GeoInformatica*, 17 (4), 635–667. doi:10.1007/s10707-012-0173-8

Gelernter, J. and Mushegian, N., 2011. Geo-parsing messages from microtext. *Transactions in GIS*, 15 (6), 753–773. doi:10.1111/j.1467-9671.2011.01294.x

Gimpel, K., *et al*.. 2010. *Part-of-speech tagging for twitter: annotation, features, and experiments*. Technical Report.Carnegie-Mellon Univ Pittsburgh Pa School of Computer Science.

Giridhar, P., *et al*.. 2015. On quality of event localization from social network feeds. *In*: *2015 IEEE International Conference on Pervasive Computing and Communication Workshops (PerCom Workshops)*, 75–80. St. Louis, MO: IEEE.

Gritta, M., Pilehvar, M.T., and Collier, N., 2018. A pragmatic guide to geoparsing evaluation. *arXiv Preprint arXiv:1810.12368*.

Guerini, M., *et al*.. 2018. Toward zero-shot entity recognition in task-oriented conversational agents. *In*: *Proceedings of the 19th Annual SIGdial Meeting on Discourse and Dialogue*, Melbourne, Australia: 317–326.

Kar, S., *et al*.. 2018. D-record: disaster response and relief coordination pipeline. *In*: *Proceedings of the 1st ACM SIGSPATIAL Workshop on Advances on Resilient and Intelligent Cities*, Seattle, WA: 13–16.

Karimzadeh, M., *et al*., 2019. GeoTxt: a scalable geoparsing system for unstructured text geolocation. *Transactions in GIS*, 23 (1), 118–136. doi:10.1111/tgis.12510

Kim, Y., 2014. Convolutional neural networks for sentence classification. *arXiv Preprint arXiv:1408.5882*.

Kumar, A. and Singh, J.P., 2019. Location reference identification from tweets during emergencies: a deep learning approach. *International Journal of Disaster Risk Reduction*, 33, 365–375. doi:10.1016/j.ijdrr.2018.10.021

Kumar, A., Singh, J.P., and Rana, N.P. 2017. Authenticity of geo- location and place name in tweets.

Li, C. and Sun, A., 2014. Fine-grained location extraction from tweets with temporal awareness. *In*: *Proceedings of the 37th international ACM SIGIR conference on Research & development in information retrieval*, Gold Coast, Queensland, Australia, 43–52.

Limsopatham, N. and Collier, N., 2016. Bidirectional LSTM for named entity recognition in Twitter messages. *COLING 2016*.

Lingad, J., Karimi, S., and Yin, J., 2013. Location extraction from disaster-related microblogs. *In*: *Proceedings of the 22nd international conference on world wide web*, Rio de Janeiro, Brazil, 1017–1020.

Maneriker, P., *et al*. 2019. A pipeline for disaster response and relief co- ordination. *In*: *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval*, Paris, France, 1337–1340.

Matsuda, K., *et al*. 2015. Annotating geographical entities on microblog text. *In*: *Proceedings of the 9th linguistic annotation workshop*, Denver, Colorado, 85–94.

Mendes, P.N., *et al*. 2011. DBpedia spotlight: shedding light on the web of documents. *In*: *Proceedings of the 7th international conference on semantic systems*, Graz, Austria, 1–8.

Middleton, S.E., *et al*., 2018. Location extraction from social media: geoparsing, location disambiguation, and geotagging. *ACM Transactions on Information Systems (TOIS)*, 36 (4), 1–27. doi:10.1145/3202662

Middleton, S.E., Middleton, L., and Modafferi, S., 2013. Real-time crisis mapping of natural disasters using social media. *IEEE Intelligent Systems*, 29 (2), 9–17. doi:10.1109/MIS.2013.126

Mikolov, T., *et al*., 2013a. Efficient estimation of word representations in vector space. *arXiv Preprint arXiv:1301.3781*.

Mikolov, T., *et al*., 2013b. Distributed representations of words and phrases and their compositionality. *In Advances in Neural Information Processing Systems*, Lake Tahoe, Nevada, 3111–3119.

Morstatter, F., *et al*. 2013. Is the sample good enough? comparing data from twitter's streaming api with twitter's firehose. *In*: *Proceedings of the International AAAI Conference on Web and Social Media*, Vol. 7, Cambridge, MA.

Ozdikis, O., Oğuztüzün, H., and Karagoz, P., 2017. A survey on location estimation techniques for events detected in Twitter. *Knowledge and Information Systems*, 52 (2), 291–339. doi:10.1007/s10115-016-1007-z

Pennington, J., Socher, R., and Manning, C.D. 2014. Glove: global vectors for word representation. *In*: *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, Doha, Qatar, 1532–1543.

Qi, P., *et al*., 2020. Stanza: a python natural language processing toolkit for many human languages. *arXiv Preprint arXiv:2003.07082*.

Reuter, C. and Kaufhold, M.-A., 2018. Fifteen years of social media in emergencies: a retrospective review and future directions for crisis informatics. *Journal of Contingencies and Crisis Management*, 26 (1), 41–57. doi:10.1111/1468-5973.12196

Ritter, A., *et al*. 2011. Named entity recognition in tweets: an experimental study. In *Proceedings of the 2011 conference on empirical methods in natural language processing*, Edinburgh, Scotland, UK, 1524–1534.

Rolnick, D., *et al*., 2017. Deep learning is robust to massive label noise. *arXiv Preprint arXiv:1705.10694*.

Sultanik, E.A. and Fink, C., 2012. Rapid geotagging and disambiguation of social media text via an indexed gazetteer. *ISCRAM*.

Tapia, A.H., Moore, K.A., and Johnson, N.J., 2013. Beyond the trustworthy tweet: a deeper understanding of microblogged data use by disaster response and humanitarian relief organizations. *ISCRAM*.

Unankard, S., Li, X., and Sharaf, M.A., 2015. Emerging event detection in social networks with location sensitivity. *World Wide Web*, 18 (5), 1393–1417. doi:10.1007/s11280-014-0291-3

Wallgrün, J.O., *et al*., 2018. GeoCorpora: building a corpus to test and train microblog geoparsers. *International Journal of Geographical Information Science*, 32 (1), 1–29. doi:10.1080/13658816.2017.1368523

Wang, J., Hu, Y., and Joseph, K., 2020. NeuroTPR: a neuro-net toponym recognition model for extracting locations from social media messages. *Transactions in GIS*, 24 (3), 719–735. doi:10.1111/tgis.12627

Weissenbacher, D., *et al*., 2015. Knowledge-driven geospatial location resolution for phylogeographic models of virus migration. *Bioinformatics*, 31 (12), i348–i356. doi:10.1093/bioinformatics/btv259

Xie, S., Wang, S., and Yu, P.S., 2016. Active zero-shot learning. *In*: *Proceedings of the 25th ACM International on Conference on Information and Knowledge Management*, Indianapolis, Indiana, 1889–1892.

Yuan, F. and Liu, R., 2018. Feasibility study of using crowdsourcing to identify critical affected areas for rapid damage assessment: hurricane Matthew case study. *International Journal of Disaster Risk Reduction*, 28, 758–767. doi:10.1016/j.ijdrr.2018.02.003

Zhou, C., *et al*., 2015. A C-LSTM neural network for text classification. *arXiv Preprint arXiv:1511.08630*.