GazPNE: Annotation-free Deep Learning for Place Name Extraction from Microblogs Leveraging Gazetteer and Synthetic Data by Rules

Preprint · September 2020

DOI: 10.13140/RG.2.2.35493.73444

Xuke Hu
German Aerospace Center (DLR)
23 PUBLICATIONS **227** CITATIONS
SEE PROFILE

Hussein Al-Olimat
Tempus Labs Inc.
17 PUBLICATIONS **124** CITATIONS
SEE PROFILE

Jens Kersten
German Aerospace Center (DLR)
32 PUBLICATIONS **315** CITATIONS
SEE PROFILE

Matti Wiegmann
Bauhaus-Universität Weimar
13 PUBLICATIONS **41** CITATIONS
SEE PROFILE

ARTICLE TEMPLATE

GazPNE: Annotation-free Deep Learning for Place Name Extraction from Microblogs Leveraging Gazetteer and Synthetic Data by Rules

ARTICLE HISTORY
Compiled September 22, 2020

Abstract
Place name extraction refers to the task of detecting precise location information in texts like microblogs. It is a vital task to assist disaster response, revealing where the damages are, where people need assistance, and where help can be found. All current approaches for extracting the place names from microblogs face crucial problems: rule-based methods do not generalize, gazetteer-based methods do not detect unknown multi-word place names, and machine learning methods lack sufficient data, which is costly to annotate on scale. We propose a hybrid method that avoids these problems, named GazPNE, which fuses rules, gazetteers, and deep learning methods to achieve state-of-the-art-performance without requiring any manually annotated data.

Specifically, we utilize C-LSTM, a fusion of Convolutional and Long Short-Term Memory Neural Networks, to decide if an n-gram in a microblog text is a place name or not. The C-LSTM is trained on 4.6 million positive examples extracted from OpenStreetMap and GeoNames and 220 million negative examples synthesized by rules and evaluated on 4,500 disaster-related tweets, including 9,026 place names from three floods: 2016 in Louisiana (US), 2016 in Houston (US), and 2015 in Chennai (India). Our method improves the previous state-of-the-art by 6%, achieving an F_1 of 0.86.

KEYWORDS
Place name extraction; Gazetteer; OpenStreetMap; Synthetic data; Rule; Microblog; Annotation free; 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, Moore, and Johnson 2013). For example, Twitter is widely used in disaster response and rescue, such as earthquakes, floods, fire, terrorist attacks, and civil unrest (Alexander 2014; Ozdikis, Oğuztüzün, and Karagoz 2017; Yuan and Liu 2018). When an emergency event occurs, extracting location information from the tweets is crucial for keeping people and authorities informed about the exact affected area, where people can receive fresh water and food, and the locations where people need rescue and medical assistance (Kar et al. 2018; Maneriker et al. 2019). However, this task is a challenge since geo-tagged tweets are sparse. According to Dutt et al. (2018), only 0.36% of the total number of tweets contain geo-tags, and these rarely reflect the mentions' exact geolocations. Thus it is necessary to extract the location information from the tweet texts (i.e., making it an extractive problem rather than a retrieval problem). This task is called Geoparsing and consists of two steps: (1) identifying references to the locations in the text, known as toponym recognition or place name tagging, and (2) geocoding, assigning geographic coordinates to the identified place name. This study focuses on the first step, which is also the most challenging part of Geoparsing, leaving the second as future work.

Currently, the approaches for extracting the place name from microblogs can be coarsely divided into three groups. The first approach is based on handcrafted rules that use pattern and regex-matching, which relies on cue words or orthographic features (Weissenbacher et al. 2015; Bontcheva et al. 2013; Malmasi and Dras 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 microblogs (Ritter et al. 2011). The second approach is gazetteer-based (Middleton, Middleton, and Modafferi 2013; Zhang and Gelernter 2014; Al-Olimat et al. 2018; Dutt et al. 2018). It finds the place names by matching a sequence of tokens with the place names in a gazetteer. The incompleteness and inaccuracy of gazetteers (e.g., OpenStreetMap) are the two main drawbacks of the gazetteer-based approaches, leading to a low recall. The third approach is statistical learning (Finkel, Grenager, and Manning 2005; Kumar and Singh 2019), tagging the place names mainly according to the contextual features of the place names, such as Stanza (Qi et al. 2020) and NeuroTPR (Wang, Hu, and Joseph 2020). However, they require a considerable amount of annotated sentences which makes those approaches ineffective in most practical situations (Guerini et al. 2018). Moreover, according to the experimental results, intrinsic features of place names are more important than contextual features in detecting the place names from microblogs due to the weak contextual cures in microblogs caused by dramatic variation of the writing styles and numerous grammar errors in microblogs (Ritter et al. 2011).

This study attempts to fuse gazetteer, rule, and deep learning-based approaches to yield a general and reliable place name extractor, named GazPNE [1] (**Gaz**etteer based **P**lace **N**ame **E**xtraction), to overcome the challenges mentioned above. The approach shares some features with recent zero-shot learning work (Xie, Wang, and Yu 2016) since it does not require any annotated sentences in the training procedure. Specifically, we obtain positive examples from gazetteers (i.e., OpenStreetMap and GeoNames). We then apply several rules to synthesize the negative examples from the positive ones and the vocabulary in Google (Mikolov et al. 2013) and Glove-embeddings (Pennington, Socher, and Manning 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). To avoid overfitting and memorizing noisy examples generated by rules, a early stopping strategy is adopted since at the early stage of training the network trends to first learn the knowledge from the examples which are "easy" to fit. These examples are normally clear examples (Zhang et al. 2016; Rolnick et al. 2017). Finally, we apply the classifier on pre-processed microblog texts to search their n-grams and select top non-overlapping candidates as the final set of detected place names.

The main contributions of this study are as follows:

(1) To the best of our knowledge, we are the first to utilize deep learning for place name extraction without requiring any manually annotated data.
(2) The proposed approach shows promising performance in robustness and generalizability compared to the current place-name extractors.
(3) We prove the feasibility and potential of fusing statistical learning and rules (expert knowledge) to compensate for the individual approaches weaknesses.

[1]The data and code can be obtained from `https://github.com/xukehu/GazPNE`

The remainder of this paper is structured as follows: In Section 2, we conduct a review of the 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 three groups. (1) Rule-based approaches, (2) Gazetteer-matching-based approaches that match the token sequence in tweets with gazetteer records, (3) Statistical Named Entity Recognition-based. Many related studies use more than one of the three approaches at different place name parsing stages. We discuss those in the paragraph that corresponds to the type of approach that contributes the most to the place name extraction task.

Handcrafted rules: Place names in sentences own certain characteristics, which could be represented by rules. For instance, Giridhar et al. (2015) used road-traffic-related Twitter to detect and locate the point-events, such as building fires. As for the event localization, the 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 the true location-identifiers, which were commonly preceded by Prepositions (IN), such as in, around, between, and after. Bontcheva et al. (2013) presented TwitIE-GATE, an open-source NLP pipeline customised to microblog text at every stage, which adapts rules from ANNIE (Cunningham et al. 2002) for the extraction of name entities. ANNIE consists of the following main processing resources: a tokenizer, sentence splitter, POS tagger, gazetteer lists, finite state transducer, orthomatcher and coreference resolver. TwitIE-GATE is evaluated on a corpus of 2,400 tweets comprising 34,000 tokens. Finally, an F1 score of 0.80 was achieved on the place name recognition task. Weissenbacher et al. (2015) proposed using a hybrid approach combining dictionary-based and rule-based heuristics for 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. The approach was tested on 300 journal articles related to phylogeography. An F1 score of 0.68 was achieved for place name extraction task. Dutt et al. (2018) proposed a system to infer location names mentioned in tweets in an unsupervised fashion. 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 phrases extracted by the above methods are then verified using a gazetteer. An F1-score of 0.79 is achieved on 1,000 randomly selected tweets.

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 variations of place names, which is also quite troublesome. Thus, it is very likely that the defined rules work well on one test data but fails on the other test data. Compared with the rule-based approaches, the gazetteer-based approach 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. Therefore, the gazetteer-based approach is now gaining more and more attention.

Gazetteer search and matching: Gazetteer is the dictionary of geospatial places with names and geo-coordinates. Popular geospatial gazetteers include OpenStreetMap, GeoNames, and the geo-tagged Wikipedia database. The usage of a few heuristics, stop words and proper nouns can 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 (Middleton, Middleton, and Modafferi 2013; Li and Sun 2014; Zhang and Gelernter 2014; Al-Olimat et al. 2018; D'Ignazio et al. 2014) have used gazetteers to extract the place names. For instance, Middleton, Middleton, and Modafferi (2013) presented a real-time crisis mapping platform capable of geoparsing tweet across multilingual datasets, which used gazetteers, street maps, and volunteered geographic information to improve geoparsing precision of street-level tweet incident reports. They stated that a high precision could be achieved in location extraction from the real-time tweets.

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. The LNEx system was evaluated on 4,500 event-specific tweets from three targeted streams achieving an F1-score of 0.81. Middleton et al. (2018) developed a geoparsing algorithm using an OpenStreetMap database and a geo-tagging algorithm using a language model constructed from social media tags and multiple gazetteers. The result showed that the OpenStreetMap database based approach can achieve an F1-scores between 0.90 and 0.97 for English and Italian tweets, respectively, and the best F1-score of 0.66 for Turkish tweets.

Gazetteer-based approaches can achieve high tagging accuracy on the test data where most of the place names are included in the gazetteers. However, it is common that many place names found in tweets are missing from gazetteers, although this can be mitigated by augmenting the gazetteer as in (Al-Olimat et al. 2018). Furthermore, many false-positives are caused due to the context-dependency of partial place names, such as 'Washington', which can be a person, sports team, or a place name depending on the textual context.

Statistical learning: As the development of machine learning, specifically deep learning, using statistical learning algorithms to extracting place names is gaining more and more attentions (Wang, Hu, and Joseph 2020; Qi et al. 2020; Gelernter and Mushegian 2011; Lingad, Karimi, and Yin 2013; Unankard, Li, and Sharaf 2015; Das and Purves 2019; Limsopatham and Collier 2016; Kumar and Singh 2019). Given abundant annotated data, statistical learning-based approaches can recognize the place names according to the context cues and the intrinsic features of place names, thus achieving a higher tagging accuracy than the rule-based approaches. Place name recognition is the subtask of named entity recognition (NER), which has been extensively investigated. Therefore, many studies (Gelernter and Mushegian 2011; Lingad, Karimi, and Yin 2013; Unankard, Li, and Sharaf 2015) leverage existing statistical-based NER models to extract place names from social media streams. For instance, Lingad, Karimi, and Yin (2013) retrained Stanford NER (Finkel, Grenager, and Manning 2005), which implements a linear chain Conditional Random Field (CRF) model to label sequences of words in text into entity types (e.g., Person, Organization, and Location). The classical NER systems are normally implemented through traditional machine learning approaches, which require manual definition and selection of the 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 the character embedding and word embedding. The model was trained on 2,349 tweets and tested on 3,850 tweets and achieved an F1-score of 0.66 on 11 entity types. The model achieved an F1-score of 0.73 on the location class. Kumar and Singh (2019) utilized a Convolutional Neural Network (CNN) based model for location extraction. The model was evaluated on 5,107 annotated tweets related to earthquakes with 6,690 place names mentions. Using 10-fold cross-validation, the model achieved an F1-score of 0.96. NeuroTPR (Wang, Hu, and Joseph 2020) extended a general Bi-LSTM architecture with several features to account for the 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 need for a large training dataset by generating annotated data from Wikipedia articles for the task of location name extraction. However, as we show in Section 4.3, the approach still suffers from the sparsity of the labels, making it achieve low recall even with the use of 599 annotated tweets and a large set of labeled locations from Wikipedia. Similar on-average-performance 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 extraction), the in-availability of sufficient 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 gazetteers 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. That excellent work has inspired our study in this paper.

In a nutshell, rule-based approaches are not generalizable and sometimes fragile. The gazetteer-based approach cannot sufficiently deal with the incompleteness of gazetteers. Traditional statistical learning-based approaches face the challenge of data sparsity, requiring abundant annotated data. Therefore, in this study, we propose utilizing a neural model to recognize the place name from tweets based only on gazetteers and synthesized data by rules. We combine the above three techniques to achieve the best performance in the location name extraction task.

3. Method

Two aspects mainly inspire the idea of this study. **(1) The challenge in detecting unknown and multi-word place names facing the gazetteer-based approach**. To recognize shorter 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 version 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, the biggest challenge lies in the recognition of multi-word place names, which normally own certain lexical and orthographic characteristics. That inspired us to use a deep learning model to capture the inherent characteristics of multi-word place names from gazetteers. By doing so, the learned model can recognize valid contractions of multi-word place names in gazetteers as well as missing ones.

(2) The combination of statistical learning with rules (or expert knowledge). Gazetteers can provide millions of entities. Therefore, this study uses only the gazetteers to train a classification model without requiring manually annotated data, which is costly to collect on scale. We obtain the positive examples from the gazetteer and manually define a couple of rules or heuristics 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 the place name (e.g., *"The of Mississippi University"*). These heuristics might be imperfect and can lead to noisy examples. However, according to (Zhang et al. 2016; Rolnick et al. 2017), the deep learning algorithm can tolerate modest amounts of noise in the training data given abundant correct training examples. Many studies (Arpit et al. 2017; Huang et al. 2019) also found that at the early stage of training, the convergence speed of the deep learning network is fast and the network trends to first learn the knowledge from the examples which are "easy" to fit. These easy examples are normally the clear examples. That provided us with 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. It consists of three main stages: (1) generating training examples, (2) training neural classifier, and (3) applying classifier in microblog text. At the training example generation stage, abundant and accurate training examples are generated. To obtain accurate positive examples, the gazetteer (i.e., OSM and GeoNames) is first filtered by removing likely misnamed items by volunteers on OSM such as tracks. Then, the gazetteer is augmented such as by replacing a token (e.g., *"road"*) with its abbreviation (e.g., *"rd"*). The negative examples are then synthesized based on the positive examples and general words by a set of rules. The generated negative examples are much more than positive examples since each positive example is used in multiple rules to generate negative examples. At the neural classifier training stage, the C-LSTM (Zhou et al. 2015) model is trained based on the collected positive and negative examples. 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. At the last stage, a microblog text is first tokenized and split into multiple subsequences by several punctuation marks. Then, the trained neural classifier is used to search the n-grams of each subsequence and select the top non-overlapping candidates with the highest estimated probability as the detected place names. n varies from 1 to 20.

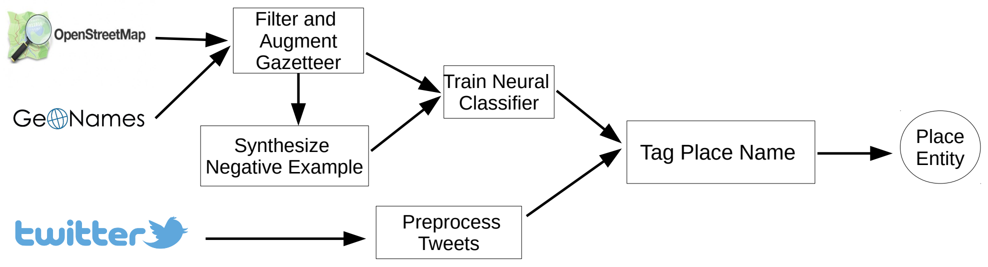


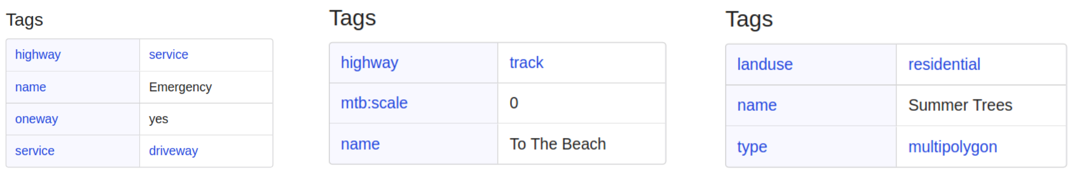
Figure 1. Workflow of our proposed place name extraction approach (GazPNE).

3.1. *Training example generation*

This study's key task is to produce as abundant and accurate training examples as possible from the gazetteer and general words. The general words are obtained from the gazetteer and the well known pre-trained word embedding, Glove (Pennington, Socher, and Manning 2014) and Google (Mikolov et al. 2013) embeddings. From the Google word embedding, the frequency of each word can also be obtained. Thus, a general word list can be produced by giving the 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 the number-prefix-suffix words (e.g., "ft","inch", and "pm") that do not appear in the gazetteer (e.g., "hwy" and "rd"). The number-prefix-suffix words are extracted from the Glove and Google embedding vocabularies, which consists of the 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 the positive examples, which consists of numbers are broken into numbers and words, such as to break *"hwy00"* to *"hwy"* and *"00"*. Apart from the general words, the location category names (e.g., "school", "road", "town") are also used to limit the negative examples. They are extracted by counting the last word of the place names in the gazetteer, and the one whose count is over a certain threshold is regarded as the location category name. Besides, all the characters in the dataset are converted to lower case, and all non-English and non-number characters are removed, such as '(!-)'.

3.1.1. Obtaining Positive Examples

1. **Filtering**: (1) Gazetteers, such as OSM, suffer from data quality issues, and thus need to be filtered to improve the accuracy of the positive examples. We rule that the items whose attributes are 'highway = footway', 'highway = service', 'highway = cycleway', 'highway = track', or 'highway = path' and the item whose


Tags

highway	service
name	Emergency
oneway	yes
service	driveway

Tags

highway	track
mtb:scale	0
name	To The Beach

Tags

landuse	residential
name	Summer Trees
type	multipolygon

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

type is 'landuse' are removed. These items are normally created and named by the volunteers themselves without referring to the mainstream maps (i.e., Google Maps and Bing maps) in which these items are not represented. For instance, Figure 2 shows three OSM items incorrectly named by the volunteers without referring to official maps. The left two items are named by their purposes. (2) Well known single words (such as *"good"*), numbers, and English alphabet letters are removed from the gazetteer if their type is not county, town, suburb, city, or state, which are marked as 'place=county', 'place=town','place=suburb','place=city', and 'place=state', respectively on OSM [2]. The well-known words are defined as the ones whose frequency is ranked among top t^f in the Google embedding vocabularies. This can reduce false positives.

2. **Augmentation**: For the place entity in the type of county, town, suburb, city, and state, the corresponding category words are removed to produce new positive samples if the category word exists, and the simplified name is not well known single words as defined above. For instance, for place name *"Jim Wells County"*, a new and valid place name is *"Jim Wells"*. The category name is added to the end of the place name for the entities in the five types without the category name. For instance, for place name *"Houston"* with the city type, a new and valid place entity is *"Houston city"*. Moreover, the place entities in the five types are given more attentions by copying them C times since they are high-frequency and important places (e.g., *"Houston"*) and often used in the microblogs. Furthermore, the country-level roads marked as 'highway = motorway' and 'highway = trunk' [3] are also emphasized in the same way since they are the most significant roads that 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, the English OSM abbreviation dictionary [4] is used to extend the gazetteer by replacing the original word with its abbreviation word.
4. **Out-of-vocabulary**: In tweets, there are some out-of-vocabulary words regarding with 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 obtain a new place name, such as *"hiagnnamalnsw street"*. This can improve the detection rate of the place names containing out-of-vocabulary words.

[2]https://wiki.openstreetmap.org/wiki/Key:place
[3]https://wiki.openstreetmap.org/wiki/Key:highway
[4]wiki.openstreetmap.org/wiki/Name_finder:Abbreviations

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 knowledge of place names in sentences and observation of positive examples. These rules are imperfect and can cause noisy negative examples. However, the issue can be mitigated by using a early stopping strategy in deep learning because the deep learning algorithm tends to fit the clear data at the early stage of training. The implementation details of these rules can be seen in the published code.

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. Thus, 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 word has changed, and the last word is not the location category name. For instance, for the place name *"National Park of New York"*, the generated two negative examples are *"Park National of New 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 it is 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 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"*, and *"on the street"* 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. **Combination of place names:** Two place names are combined to form new negative examples if it is not included in the positive examples, such as *"west little creek florence"* from *"west little creek"* and *"florence"*.

10. **Insert general words between place names:** A general word is inserted between two place names to generate new negative examples if it is 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 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 network(RNN) are the better option since they specialize in sequential modeling. CNN-based models are better for non-sequential place names, since they focus on local response features. In this study, we apply the C-LSTM (Zhou et al. 2015) model for classifying place entities, which combines a CNN and a LSTM to achieve the best of both. In C-LSTM, a CNN is first applied to extract higher-level sequences of word features. 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 core is a CNN layer with different features followed by an LSTM layer and a single dropout layer between the LSTM layer and the output layer (a softmax layer). Note that three types of feature vectors for a word are concatenated in the bottom layer, which will be introduced in Section 3.2.2.

3.2.2. Classifier features

We use three kinds of features in the classifier, following Guerini et al. (2018). The input layer concatenates all the features in a single vector. Specifically, the three types are two different word embeddings (i.e., generic and specific), and six handcrafted features. The generic word embedding is the pre-trained GloVe 50-Dimensional Word Vectors to capture generic language use. The second word embedding is employed to capture language use that is specific for each domain. We obtained that using the word2vector algorithm on the positive examples, with a dimension of 30, a minimum word frequency cutoff of 1, and a window size of 2. The six handcrafted features we used to represent the key structure of a example explicitly 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 with reference to 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 86, while the dimension of the example vector equals 1720.

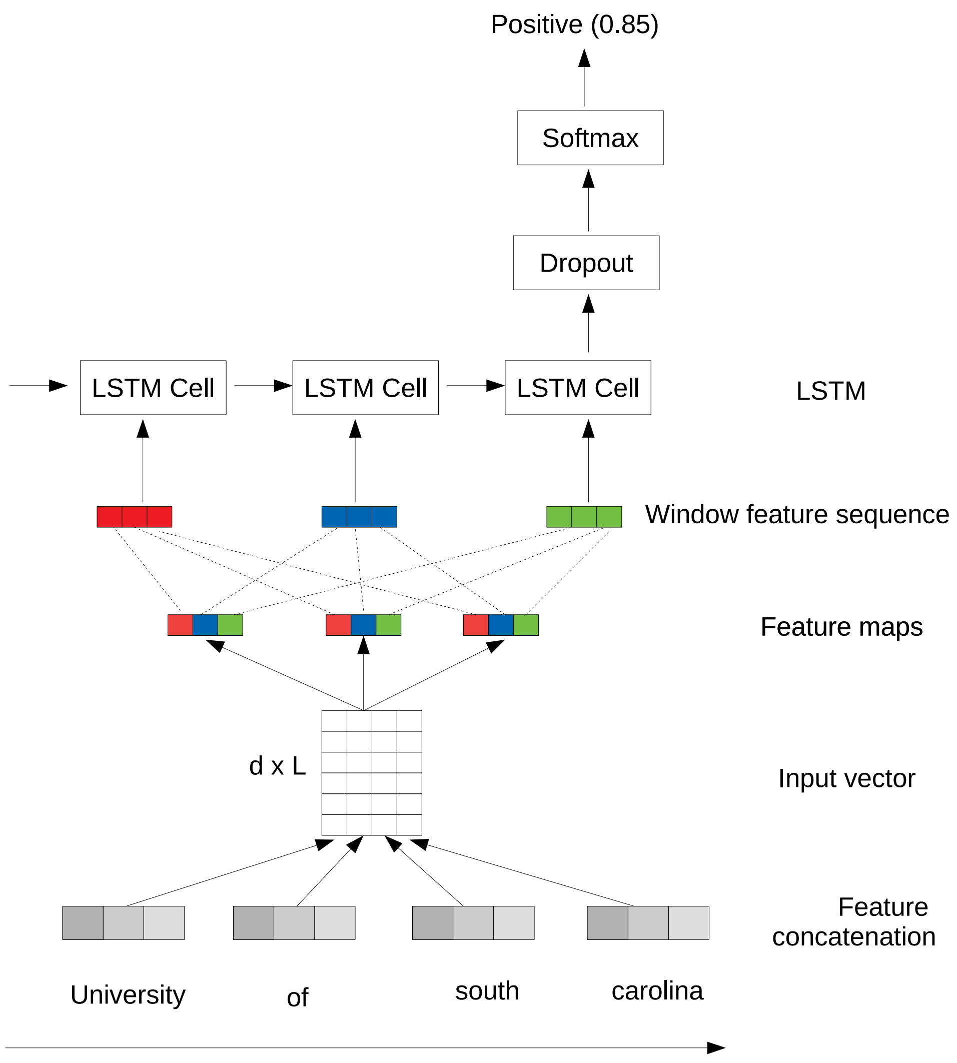


Figure 3. The architecture of the C-LSTM model for place classification. In the layer of feature concatenation, four types of features are concatenated, forming the next layer's input vector. L denotes the number of words in the input entity while d denotes the dimension of the word vector. Blocks of 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. The last hidden unit of LSTM is used as the input of the dropout layer, and a softmax layer is the output layer.

3.3. *Place extraction from tweets*

3.3.1. Tweet preprocessing

Given a tweet text, it is first cleared and segmented into sub sequences. Specifically, the retweet handles, URLs, non-ASCII characters, and all user mentions in the tweet are removed. 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"*. The misspelled tokens are then corrected by using the Symmetric Delete Spelling Correction algorithm (SymSpell)[5]. The out-of-vocabulary words are replaced with the pre-defined word (such as 'hiagnnamalnsw') in the training samples generation procedure. 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"*. Last, stop words including [ ] , % () ! ; : < > . are used to segment the tweet text into sub sequences. For instance, *"Driving in from Sugar Land! Any roads to avoid?"* are splited into two sub sequences *"Driving in from Sugar Land"* and *"Any roads to avoid"*.

3.3.2. Place name tagging

We used the classifier to calculate the confidence score of each subset of a subsequence from the raw tweet. Then, we select as candidates the ones whose score is over a certain threshold denoted by t^s, which are then ranked in descending order according to the confidence score. Next, the ranking list is adjusted by moving the subset to the ahead of the one it includes if the score of the former is lower than that of the latter. That is, if candidate a (e.g., *"west florence"*) includes candidate b (e.g., *"florence"*), a is moved to the ahead 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. For instance, given a subsequence *"Flood is serious at Little Creek West Florence"*, the *Little Creek* (0.9) | *Little Creek West* (0.88) | *Florence* (0.97)| *West Florence* (0.95) are first 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. They are indicators of a place name but are unnecessary for a place name. The pseudo-code of the place name tagger is provided in Algorithm 1.

[5]https://github.com/wolfgarbe/symspell

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:
        $Pla$ // place names in the sentence;
1:  procedure TAGGER
2:      $S_{can} \leftarrow null; P_{can} \leftarrow null; Pla \leftarrow null$
3:      generate all the sub sets of $S$ and saved into $S_{sub}$
4:      for $s \in S_{sub}$ do
5:          $p \leftarrow M.calProbability(s)$ // calculate positive probability
6:          if $p > s^t$ then
7:              $S_{can} \leftarrow S_{can} \cup \{s\}$; $P_{can} \leftarrow P_{can} \cup \{p\}$
8:      sort $S_{can}$ according to $P_{can}$ in descend order and saved to $T$
9:      $l \leftarrow len(T); index \leftarrow 0$
10:     while $index < l$ do
11:         $s \leftarrow T[index]$
12:         search $T[0 : index - 1]$ to get first index ($k$) of child of $s$
13:         if $k >= 0$ then
14:             update $T$ by moving $T[index]$ ahead of $T[k]$
15:         $index \leftarrow index + 1$
16:     for $s \in T$ do
17:         if $s$ not overlap any element of $Pla$ then
18:             $Pla \leftarrow Pla \cup \{s\}$
19:     for $l \in Pla$ do
20:         if $l[-1] \in area$ then
21:             delete $l[-1]$ and update $Pla$
22:     return $Pla$
```

4. Experiments

4.1. *Training data preparation*

We use the tools provided by OSMNames[6] to extract the OSM data of the south of the US with the boundary box [-104.79, 29.57, -74.5, 40.31], representing the minimum longitude and latitude and maximum longitude and latitude coordinates, respectively. It includes the Houston and Louisiana area. Moreover, the OSM data of the south area of India with the boundary box [73.59, 8.58, 82.76, 20.47] is also extracted, which includes the Chennai area. The chosen boundary boxes are shown in Figure 4. Besides, the place names in the type of libraries, airports, universities, hospitals, schools, churches, parks, hotels, and theaters across the US and India are extracted from GeoNames since they are important public places but not sufficiently provided by OSMNames. Last, a couple of places are manually added into the positive examples, including *'Chennai'*, *'India'*, and , *'United States'*, and the abbreviation of states, including *'tx'* for *'Texas'*, *'MS'* for *'Mississippi'*, and *'La'* for *'Louisiana'*. They are marked on OSM but cannot be directly obtained from OSMNames due to some technical problems. In total, 4,688,163 initial positive examples are obtained from the gazetteers, which are then augmented and extended to 22,603,449. Based on the augmented positive examples, the negative examples are then synthesized. In total, 221,637,424 negative examples are generated. Specifically, 15,621,481, 1,514,161, 123,326,263, 17,782,834, 36,149,912, 20,040,794, 276,724, 1,915,676, 4,999,429, and 2,000,000 negative examples are generated by applying the defined ten rules in section 3.1.2, respectively.

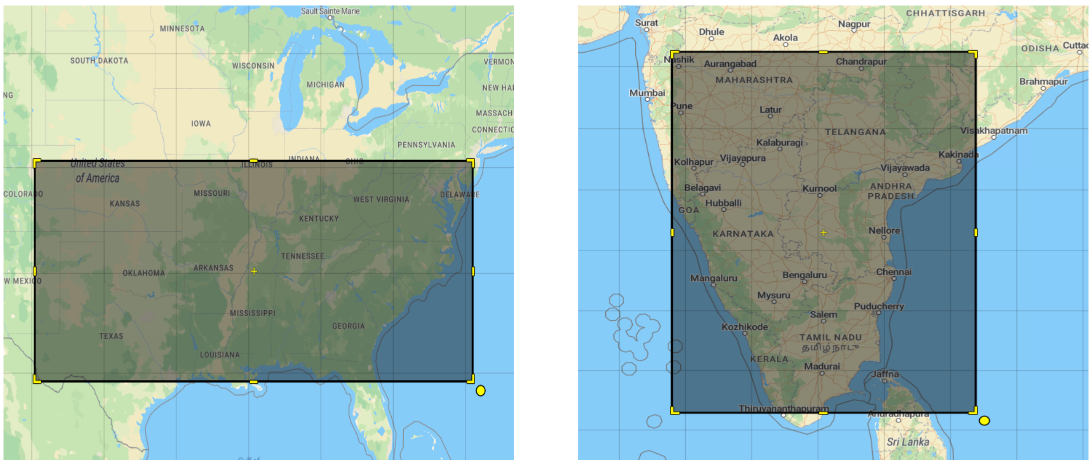


Figure 4. Boundary box for extracting the OSM data. Shallow rectangle denotes the boundary box. Left figure shows the boundary box for the US area and right figure shows the boundary box for the India area.

There are several key parameters for the proposed approach. During the procedure of collecting training examples, the count threshold for the location category name (t^l) is set to 500, 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 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) set to 120, 1, 120, and 0.5, respectively. During the place name tagging procedure, the score

[6]https://osmnames.org/

threshold (t^s) for choosing the valid candidate is set to 0.84. The sensitivity analysis of key parameters such as the score threshold and filter size on the tagging accuracy is also conducted. Moreover, we compare the C-LSTM with the multi-channel CNN (Kim 2014) and Bi-LSTM (Guerini et al. 2018) based NER systems.

4.2. *Test data*

To demonstrate the effectiveness of the proposed place name extractor, we used the 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[7]. We found a few missing location mentions in the annotated data due to human error. 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 so we can directly compare our results with the results previously reported in the paper.

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 Lousiana flood, and ambiguous locations (e.g., 'my house') . In the Louisiana dataset, there are 2,918 (66% inLOC, 13% outLOC, and 22% ambLOC) annotated place names. In the Houston dataset, there are 4,177 (66% inLOC, 7% outLOC, and 27% ambLOC) annotated place names. In the Chennai dataset, there are 4,589 (75% inLOC, 4% outLOC, and 21% ambLOC) annotated place names.

4.3. *Tagging result*

We compare our approach GazPNE with competitive place name extraction systems, including Google NLP, Stanza, CLIFF, NeuoTPR, CNN-based approach (Kumar and Singh 2019), and LNEx on the same testing dataset. We considered the entities of type Location, Organization, or Address as location mentions from Google NLP[8]. We used the NER tool in Stanza[9] and kept the entities of type Location, Organization, Facility, and Geopolitical Entity. As for CLIFF[10], 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). For NeuroTPR we used their model and implementation to tag the evaluation dataset[11]. The CNN-based approach (Kumar and Singh 2019) 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 the test data and the remaining nine groups are used as the training data. The average result of the ten tests are 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

[7]The data can be obtained from https://rebrand.ly/LocationsDataset

[8]We recently tested Google's NLP API on the same datasets, hence the difference in the scores in our paper and the LNEx paper (Al-Olimat et al. 2018).

[9]https://stanfordnlp.github.io/stanza/

[10]https://cliff.mediacloud.org/

[11]We did not retrain the model, we simply used the model provided by the authors at https://github.com/geoai-lab/NeuroTPR

one (e.g., Chennai) as the test data, named transfer CNN. This is to evaluate the transferability of the trained model.

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"). Table 1 contains the full results of the different approaches.

Table 1. Tagging result of multiple place name extractors

	Louisiana			Houston			Chennai			Avg
	P	R	F	P	R	F	P	R	F	F
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
CLIFF	0.82	0.79	0.80	0.80	0.49	0.61	0.74	0.37	0.49	0.63
NeuroTPR-1	0.83	0.48	0.61	0.77	0.28	0.42	0.81	0.32	0.46	0.50
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	0.89	**0.85**	**0.87**	**0.88**	**0.79**	**0.84**	**0.91**	**0.82**	**0.87**	**0.86**

On all of the three datasets, our proposed approach achieves the best F1-score of 0.87, 0.84, and 0.87, respectively. The second-best performing approach was LNEx (Al-Olimat et al. 2018), achieving the F1 scores of 0.82, 0.76, and 0.85, respectively. 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), which is not expected, especially that the method was designed for Twitter texts. The recall was the second lowest among all the tools, suggesting that this technique is struggling with generalizing across different datasets and that it would need much more data to be competitive with the other techniques. The 10-fold CNN achieves an acceptable tagging result. However, it requires that the test data sets should be similar to the 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 the chosen training datasets and has a poor transfer-ability. This can be proved by the low tagging performance achieved by the transfer CNN. Chennai dataset is different with 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 the Chennai datasets by the model.

Compared to the gazetteer-based approach, our method achieved higher precision and recall across the board (except for the precision on the Louisiana 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 can learn the characteristics of multi-word place names such that many multi-word place names not included in the gazetteer can be detected, such as *"Chennai Airport"*, *"Keith Weiss Park"*, *"Hidden Valley Church of Christ"*, *"emergency operation center"*, and *"Paint Creek Bridge"*. The second is that it can correctly judge which is not likely a place name after learning from abundant negative training examples generated by rules. Moreover, it is able to tolerate the modest amounts of noise in the training examples (Rolnick et al. 2017) caused by inaccurate rules. This has led to the high precision among all the three datasets. These two aspects 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 5. The detection rate of the one-word place name is the highest on all of the three datasets. The other types of place names' detection rates are all over 0.5 except the 4-grams in Houston dataset, showing an acceptable performance of detecting multi-word place names. In the future, we will investigate how to further enhance the detection of multi-word place names.

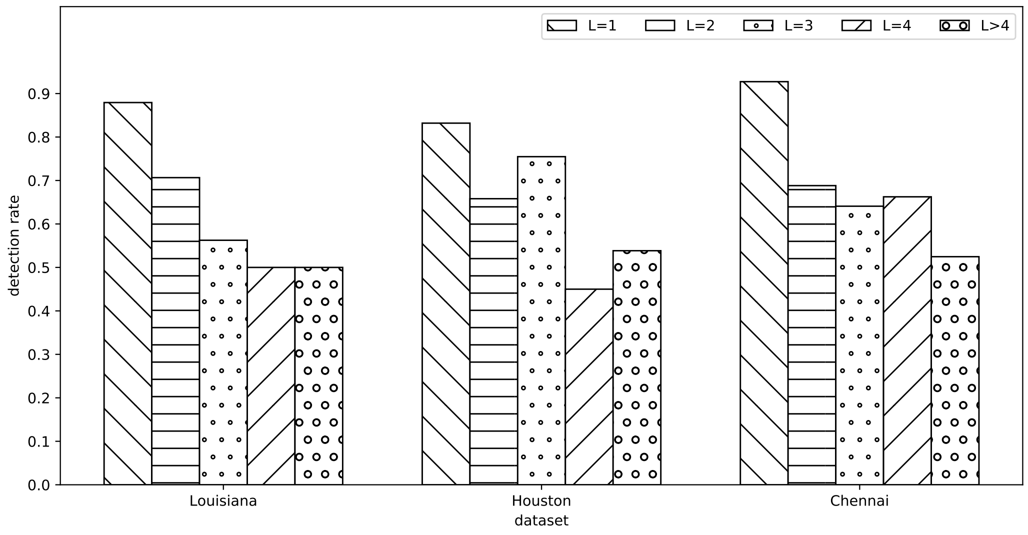


Figure 5. Detection rate of the place names with variant length on three datasets.

Furthermore, we analyzed the cases of incorrect detection by GazPNE. We choose 17 tweets from the dataset and present the tagging result by the proposed approach, which can be seen from Table 4. Generally, the incorrect detection has the following seven types. (1) Many person names are detected as place names, such as “Katy” and “Nelson” in the 5th tweet because they appear in the gazetteer. This issue can be solved by removing the person names from the text with the help of a person name dictionary before tagging the text as Middleton et al. (2018) did. (2) Some one-word place names which are not included in gazetteer cannot be recognized, such as “Binz” in the 2nd tweet, “khou11” in the 6th tweet, “hou” in the 7th tweet, and “thoraipakkamm” in the 14th tweet. We found 291 such place names, which are unseen abbreviations or nicknames (e.g.,“hou”, “nws”, “htown”, and “kou11”) in the Houston dataset, while there are 201 “hou” which is the abbreviation of “houston”. They are not presented in the gazetteers. The correct detection of them can increase the F1-score for the Houston dataset to 0.90. (3) The long place name that does not own the lexical and structural characteristics cannot be detected if they are not included in the gazetteer, such as “pondy bazaar” in the 14th tweet. (4) The erroneous break of the hashtags by the statistical word segmentation algorithm (Matsuda et al. 2015). The used algorithm breaks “#lawx” into “law” and “x” since this combination is more probable, but the correct break should be “la” and “wx”. The proposed algorithm is able to detect “la”, but the incorrect 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 0.87 to 0.89. (5) Some one-word place names (such as “Spring”, “269”) have been filtered from the gazetteer because they are the 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”. (7). Incomplete detection or over-detection of the place names, such as “Chennai beach station” in the 15th tweet detected as “Chennai beach”, and “Memorial Drive” in the 11th tweet detected as “flooded Memorial Drive”. Some of these issues can be mitigated by combining the contextual features from tweet texts and the intrinsic features from gazetteers. This will be investigated in the future.

Furthermore, we analyzed the effectiveness of several tricky strategies.

Out-of-Vocabularies: It 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-vocabularies 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 detected place names containing certain meaningless words in end of the place names. 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, *“Meyerland area”*, which is the detected place name by the proposed approach and *’area’* is then removed.

4.4. *Sensitivity analysis*

4.4.1. Impact of score threshold on tagging accuracy

The score threshold (t^s) for choosing the valid candidate during the place name tagging stage is significant. 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 6 shows the result achieved by the proposed approach as the change of t^s. Typically, a lower t^s value would lead to a higher false-positive result, while a higher t^s value would lead to a lower true-positive result. We can see when t^s is set in the range of [0.82,0.89], the highest F1-score is achieved. The lowest F1-score of 0.85 is achieved when t^s is set to 0.5. The results reveal that t^s would affect the tagging result, but in general, the performance remains stable as the change of t^s.

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

The impact of the 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 7 to avoid overfitting or memorizing noisy examples. During the training procedure, 10,000,000 examples are used as the test set. The test accuracy of different filter length configurations is shown in Table 2.

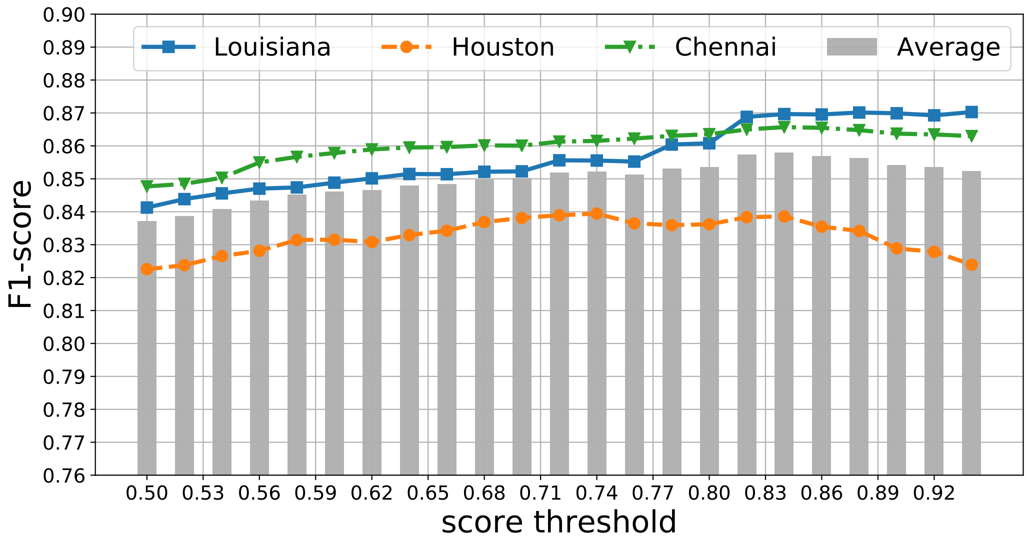


Figure 6. Impact of t^s on the F1-score for three datasets.

We can see the filter length of 1 is the best configuration for the test accuracy (at 0.9946), 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 quite close. Therefore, they are further evaluated and compared by applying them in extracting the place name from the three tweet datasets. In the experiment, 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 2 with the best t^s value in the fourth column. Still, the filter length of 1 performs the best. This is because most of the place names in the gazetteer are short. Figure 7 shows the length distribution of the place names from gazetteers, and most of them lie in the range of [1,4]. Therefore, a small filter length can achieve better performance.

Table 2. Impact of filter size on model performance

filter length	test accuracy	average F1 score	best t^s
$f^l = 1$	0.9946	0.86	0.84
$f^l = 2$	0.9937	0.85	0.90
$f^l = 3$	0.9936	0.85	0.88
$f^l = 4$	0.9931	0.85	0.88

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

In this study, we adopt the C-LSTM model, which combines the CNN and LSTM. We hope it can achieve the best of both worlds. To verify this, 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 as 3, [1,2,3], 120, and 0.5, respectively. The

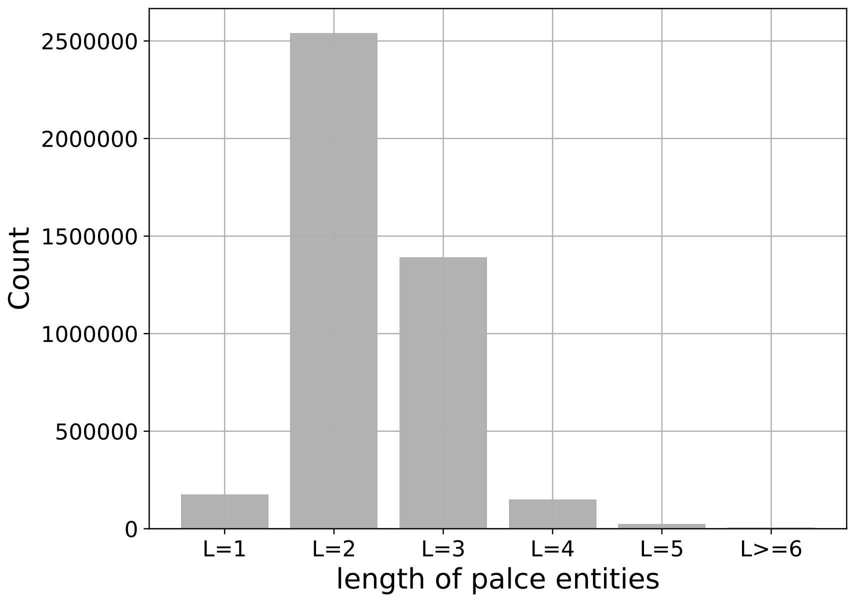


Figure 7. Distribution of length of place entities in gazetteers.

key parameters for Bi-LSTM include the memory dimension and the dropout rate, which are set as 120 and 0.5, respectively. The training epoch is set to 7. 10,000,000 examples are used as the test set. The achieved test accuracy of the three models is shown in Table 3. We can see C-LSTM and CNN outperform Bi-LSTM in the test accuracy. This is mainly because most of the place entities are short that the lexical characteristics dominate the structural characteristics of place entities. Moreover, six structural features are manually defined, which can already represent the structural characteristics of the place entities to certain extends. This is why CNN performs better than Bi-LSTM based model. Furthermore, the trained models are used to extract the 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 are shown in the third column of Table 3 with the best t^s in the fourth column. Still, C-LSTM performs the best, while Bi-LSTM performs the worst.

Table 3. Comparison of different models

model	test accuracy	average F1 score	best t^s
C-LSTM	0.9946	0.86	0.84
CNN	0.9930	0.85	0.90
Bi-LSTM	0.9406	0.80	0.84

5. Discussions

Inaccurate rules and noisy data: In this study, the negative training examples are generated by a couple of heuristics or explicit rules, which would lead to incorrect examples. For example, according to the rule, negative example *"christian college of kansas"* is generated by inserting *"christian"* before *"college of kansas"*. No spatial entity 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. The deep learning algorithm can learn the characteristics of the place names based on the majority of correct training examples. Thus, *"st john united methodist church"* is classified as positive by the learned model, although it is not in the training data, proving the capability of deep learning in tolerating noisy data to some degrees. However, it is still unclear if the noisy synthesized data have impacted the model, whether the deep learning model is tolerant of the imperfect rules, or whether the trained deep learning model can improve the rules or not. These issues are significant since noisy data, non-abundant training data, and weak interpretation are the biggest challenge faced in deep learning approaches. The combination of rules and deep learning in this study has inspired us to investigate these issues in the further.

Geoparsing: As mentioned before, this study focuses on place name tagging, which is the first step of location information extraction from tweets. The second step is geoparsing, which is to determine the latitude and longitude coordinates of the detected place name. That is necessary and will be our future focus. In this step, the gazetteer or map API (such as Google Map API) will be searched to find the place name's coordinates. However, the biggest challenge lies in two aspects. The first is the numerous ambiguities existing between different places, such as Manchester, NH USA versus Manchester, UK (geo ambiguities). 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. The important places are frequently used in social media sites and should be given a higher confidence score. Moreover, the location information in the user profile, the frequently mentioned place in the same time period which is very likely the location of the events. The places in the same tweet text or in the tweet text of the same semantic cluster are also strong features about the correct match. Therefore, to overcome this challenge, a semi-supervised machine learning approach can be developed, which requires only a few annotated data and adds new training examples from unannotated data by using co-training. The second is that many detected place names are not included in the gazetteer or the map API, which is normally informally presented but known by the locals. No existing solutions can solve this issue. This will be investigated in the future.

Cross-country application: The place name extractor is locally applicable that it only works for southeast US and south of India since only the place names in these areas have been used. How to make the model globally applicable is a challenge considering the huge number of place entities in a global gazetteer that can cause low-efficient training and the cross-linguistic issue. To solve the first issue, we can first obtain the administrative levels (e.g., country, state, city, suburb, town, country, and road) of place entities. Then, the entities of type country, region, city, town, and country are collected while the others of type road, park, river, etc. are partially collected. The former is much smaller than the latter, and the former is normally non-sequential that can only be recognized by their lexical features. The latter is huge and normally sequential. Only partial entities are needed, through which the model can learn the structural characteristics of these place entities. By doing so, the required data is dramatically reduced without the sacrifice of the recognition performance. To solve the cross-language issue, the gazetteers in the other countries (such as Germany and Italy) will also be utilized and processed in the same way as we do in this study. Then, a cross-lingual word embeddings (such as the pre-trained BERT (Devlin et al. 2018) model) would be used and a cross-lingual classification model can be then learned.

Tweets in different languages or in mixed languages will be used to evaluate the cross-lingual capability of the proposed strategy.

6. Conclusion

We proposed a deep learning-based place name extraction approach, relying only on gazetteers without requiring any manually annotated data. It combines gazetteers, rules, and deep learning into a hybrid approach to achieve the best of all. The approach was evaluated on three public tweet datasets with 9,026 place entities in total. An average F1-score of 0.86 is achieved, which is much higher than several competitive approaches on the same dataset. The applied C-LSTM model slightly outperforms the classic multi-channel CNN. Bi-LSTM performs the worst since the place names are short, and the lexical features dominate the structural features. The study introduces a new way of combining rules (prior knowledge) with deep learning, which produces abundant training examples by rules. This can be applied to other domains. In the future, we plan to add adversarial noise into the training examples and combine the contextual cues of a place name from sentences to further improve the accuracy and robustness of the proposed approach.

References

Al-Olimat, Hussein, Krishnaprasad Thirunarayan, Valerie Shalin, and Amit Sheth. 2018. "Location Name Extraction from Targeted Text Streams using Gazetteer-based Statistical Language Models." *Proceedings of the 27th International Conference on Computational Linguistics* 1986–1997. https://www.aclweb.org/anthology/C18-1169.

Alexander, David E. 2014. "Social media in disaster risk reduction and crisis management." *Science and engineering ethics* 20 (3): 717–733.

Arpit, Devansh, Stanisław Jastrzebski, Nicolas Ballas, David Krueger, Emmanuel Bengio, Maxinder S Kanwal, Tegan Maharaj, et al. 2017. "A closer look at memorization in deep networks." *arXiv preprint arXiv:1706.05394* .

Bontcheva, Kalina, Leon Derczynski, Adam Funk, Mark A Greenwood, Diana Maynard, and Niraj Aswani. 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*, 83–90.

Cunningham, Hamish, Diana Maynard, Kalina Bontcheva, and Valentin Tablan. 2002. "GATE: an architecture for development of robust HLT applications." In *Proceedings of the 40th annual meeting on association for computational linguistics*, 168–175. Association for Computational Linguistics.

Das, Rahul Deb, and Ross S Purves. 2019. "Exploring the Potential of Twitter to Understand Traffic Events and Their Locations in Greater Mumbai, India." *IEEE Transactions on Intelligent Transportation Systems* .

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

D'Ignazio, Catherine, Rahul Bhargava, Ethan Zuckerman, and Luisa Beck. 2014. "Cliff-clavin: Determining geographic focus for news articles." *NewsKDD: Data Science for News Publishing, at KDD 2014* .

Dutt, Ritam, Kaustubh Hiware, Avijit Ghosh, and Rameshwar Bhaskaran. 2018. "SAVITR: A system for real-time location extraction from microblogs during emergencies." In *Companion Proceedings of the The Web Conference 2018*, 1643–1649.

Finkel, Jenny Rose, Trond Grenager, and Christopher Manning. 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. Association for Computational Linguistics.

Gelernter, Judith, and Nikolai Mushegian. 2011. “Geo-parsing messages from microtext.” *Transactions in GIS* 15 (6): 753–773.

Giridhar, Prasanna, Tarek Abdelzaher, Jemin George, and Lance Kaplan. 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. IEEE.

Guerini, Marco, Simone Magnolini, Vevake Balaraman, and Bernardo Magnini. 2018. “Toward zero-shot entity recognition in task-oriented conversational agents.” In *Proceedings of the 19th Annual SIGdial Meeting on Discourse and Dialogue*, 317–326.

Huang, Jinchi, Lie Qu, Rongfei Jia, and Binqiang Zhao. 2019. “O2u-net: A simple noisy label detection approach for deep neural networks.” In *Proceedings of the IEEE International Conference on Computer Vision*, 3326–3334.

Kar, Shruti, Hussein S Al-Olimat, Krishnaprasad Thirunarayan, Valerie L Shalin, Amit Sheth, and Srinivasan Parthasarathy. 2018. “D-record: Disaster Response and Relief Coordination Pipeline.” In *Proceedings of the 1st ACM SIGSPATIAL Workshop on Advances on Resilient and Intelligent Cities*, 13–16.

Kim, Yoon. 2014. “Convolutional neural networks for sentence classification.” *arXiv preprint arXiv:1408.5882* .

Kumar, Abhinav, and Jyoti Prakash Singh. 2019. “Location reference identification from tweets during emergencies: A deep learning approach.” *International journal of disaster risk reduction* 33: 365–375.

Li, Chenliang, and Aixin Sun. 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*, 43–52.

Limsopatham, Nut, and Nigel Collier. 2016. “Bidirectional LSTM for named entity recognition in Twitter messages.” *COLING 2016* .

Lingad, John, Sarvnaz Karimi, and Jie Yin. 2013. “Location extraction from disaster-related microblogs.” In *Proceedings of the 22nd international conference on world wide web*, 1017–1020.

Malmasi, Shervin, and Mark Dras. 2015. “Location mention detection in tweets and microblogs.” In *Conference of the Pacific Association for Computational Linguistics*, 123–134. Springer.

Maneriker, Pranav, Nikhita Vedula, Hussein S Al-Olimat, Jiayong Liang, Omar El-Khoury, Ethan Kubatko, Desheng Liu, et al. 2019. “A Pipeline for Disaster Response and Relief Coordination.” In *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval*, 1337–1340.

Matsuda, Koji, Akira Sasaki, Naoaki Okazaki, and Kentaro Inui. 2015. “Annotating geographical entities on microblog text.” In *Proceedings of the 9th linguistic annotation workshop*, 85–94.

Middleton, Stuart E, Giorgos Kordopatis-Zilos, Symeon Papadopoulos, and Yiannis Kompatsiaris. 2018. “Location extraction from social media: Geoparsing, location disambiguation, and geotagging.” *ACM Transactions on Information Systems (TOIS)* 36 (4): 1–27.

Middleton, Stuart E, Lee Middleton, and Stefano Modafferi. 2013. “Real-time crisis mapping of natural disasters using social media.” *IEEE Intelligent Systems* 29 (2): 9–17.

Mikolov, Tomas, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. “Distributed representations of words and phrases and their compositionality.” In *Advances in neural information processing systems*, 3111–3119.

Ozdikis, Ozer, Halit Oğuztüzün, and Pinar Karagoz. 2017. “A survey on location estimation techniques for events detected in Twitter.” *Knowledge and Information Systems* 52 (2): 291–339.

Pennington, Jeffrey, Richard Socher, and Christopher D Manning. 2014. “Glove: Global vectors for word representation." In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, 1532–1543.

Qi, Peng, Yuhao Zhang, Yuhui Zhang, Jason Bolton, and Christopher D Manning. 2020. "Stanza: A python natural language processing toolkit for many human languages." *arXiv preprint arXiv:2003.07082* .

Ritter, Alan, Sam Clark, Oren Etzioni, et al. 2011. "Named entity recognition in tweets: an experimental study." In *Proceedings of the 2011 conference on empirical methods in natural language processing*, 1524–1534.

Rolnick, David, Andreas Veit, Serge Belongie, and Nir Shavit. 2017. "Deep learning is robust to massive label noise." *arXiv preprint arXiv:1705.10694* .

Tapia, Andrea H, Kathleen A Moore, and Nichloas J Johnson. 2013. "Beyond the trustworthy tweet: A deeper understanding of microblogged data use by disaster response and humanitarian relief organizations." In *ISCRAM*, .

Unankard, Sayan, Xue Li, and Mohamed A Sharaf. 2015. "Emerging event detection in social networks with location sensitivity." *World Wide Web* 18 (5): 1393–1417.

Wang, Jimin, Yingjie Hu, and Kenneth Joseph. 2020. "NeuroTPR: A neuro-net toponym recognition model for extracting locations from social media messages." *Transactions in GIS* .

Weissenbacher, Davy, Tasnia Tahsin, Rachel Beard, Mari Figaro, Robert Rivera, Matthew Scotch, and Graciela Gonzalez. 2015. "Knowledge-driven geospatial location resolution for phylogeographic models of virus migration." *Bioinformatics* 31 (12): i348–i356.

Xie, Sihong, Shaoxiong Wang, and Philip S Yu. 2016. "Active zero-shot learning." In *Proceedings of the 25th ACM International on Conference on Information and Knowledge Management*, 1889–1892.

Yuan, Faxi, and Rui Liu. 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.

Zhang, Chiyuan, Samy Bengio, Moritz Hardt, Benjamin Recht, and Oriol Vinyals. 2016. "Understanding deep learning requires rethinking generalization." *arXiv preprint arXiv:1611.03530* .

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

Zhou, Chunting, Chonglin Sun, Zhiyuan Liu, and Francis Lau. 2015. "A C-LSTM neural network for text classification." *arXiv preprint arXiv:1511.08630* .

Table 4. Examples of tweets and tagging results of the proposed approach. Bond text are the true place entity in the tweet. Underline texts are the incorrectly detected place entities.

Tweet	Extracted place entity
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/QQI8XI8OdX	(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), (flooded 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)
#**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)
RT @PIB_India: #**Chennai**Floods: Railway Shuttle service from **Chennai beach station** to **Arakkunnam station** in every 45 minute.	(Chennai), (Chennai beach), (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)