Rapid Geotagging and Disambiguation of Social Media Text via an Indexed Gazetteer

Evan A. Sultanik
Johns Hopkins University APL
Evan.Sultanik@jhuapl.edu

Clayton Fink
Johns Hopkins University APL
Clayton.Fink@jhuapl.edu

ABSTRACT

Microblogging services like Twitter afford opportunities for real time determination of situation awareness during crises as people report, via their statuses, information about events on the ground. An important component of the information included in a tweet are mentions of place names that may be sites of damage, injuries, or relief efforts. Methods for extracting these place names and determining the actual location being referenced are an essential part of the suite of tools required for automated extraction of situation awareness from tweets. Extracting and disambiguating place name mentions from text have been areas of extensive research. Twitter, however, presents challenges given the 140 character restriction on status and the informal, abbreviated language that are a norm in this communication channel. Named entity recognizers, which are dependent on labeled training data, may not be useful in this medium for extracting location mentions because the typical training domains for these taggers are absent the noise found in Twitter statuses. Additionally, the contextual information that is necessary for disambiguating place names is not always present. In this paper, we demonstrate a new technique, RapidGeo, for extracting and disambiguating place names from a location specific Twitter feed using an unsupervised technique for tagging location mentions and relying on the known geographic context of the feed for disambiguation. Our location tagging technique performs much better than an off-the-shelf named entity recognizer and we achieve reasonable precision in disambiguating extracted place names. We argue that such fast, high precision, unsupervised approaches are needed when important, actionable information is required from noisy data sources such as Twitter.

Keywords

Geolocation, Disambiguation, Twitter, Gazetteer

INTRODUCTION

On April 27, 2011 a tornado hit Tuscaloosa, Alabama causing massive damage and resulting in 47 deaths and 1000 reported injuries. In the hours and days after the disaster, social media sites such as Facebook and Twitter were put to use for brokering information about the damage to the city, the status of relief efforts, as well as a forum for expressions of concern and grief. These channels are playing an increasingly large role in the public reaction to natural disasters and those responsible for emergency management are increasingly interested in how they might be used to improve situation awareness during such events.

Much of the recent research work done in addressing this need has focused on Twitter. Twitter has certain advantages that make it a good model for such research. Among these is the ability to query tweets from a particular location, making it possible to focus on tweets from users directly affected by a disaster. The ubiquity of cell phones and smart phones, and the increasing portion of social media content coming from such platforms, increases the utility of such location-based data streams for gauging the reaction to an event. There have been significant contributions in describing how Twitter is used during crisis and how information brokering (retweets, sharing links) increases during crises (Hughes and Palen 2009), and distinguishing between how Twitter users proximate to a disaster share information differently than those remote from it (Starbird and Palen 2010). There has also been work on using natural language processing to distinguish rumor from fact in Twitter feeds (Mendoza and Poblete 2010) and developing

classifiers for extracting tweets that contain situation awareness information (Verma *et al.* 2011). One issue that has not been extensively addressed is the extraction and disambiguation of location mentions in tweets. Many of the statuses posted in response to a crisis contain references to locations. People may be reporting instances of damage, the sites of relief operations, or asking about the conditions in a specific neighborhood. Thus, being able to accurately identify and disambiguate location references in tweets is an essential part of automated approaches for extracting situation awareness information from tweets.

Tweets exemplify the phrase "language in the wild." Users abbreviate words, freely use web slang ("OMG", "SMH", "LOL", *etc.*), capitalize unpredictably, and frequently misspell words. Complete, well formed sentences are also not the norm. Named entity recognizers (NER) are usually trained on traditional corpora such as newswire reports that represent much more formal uses of written language. Obtaining reliable annotations of text for training NER taggers is expensive and difficult even for traditional corpora, much less noisy social media text. Domain shift issues also exist for text corpora, making the challenge of using supervised techniques even greater. Using automated techniques for extracting location references from Twitter data during a crisis via supervised learning—even if using training data from Twitter during a previous, yet similar incident—may not be feasible. Robust unsupervised techniques, we argue, are required that can work across incidents without the requirement of training data.

Extracting entity mentions referencing a location is only a first step, however, and techniques for disambiguating location mentions, or *toponyms*, are needed. Most approaches for toponym disambiguation rely on the context of the mention in relation other mentions in a document. For example, if a document contains mentions of "Laurel", "Columbia", and "Brooklyn", the toponym "Brooklyn" would likely be interpreted as Brooklyn, Maryland rather than the borough of New York City, given that the other topynyms in the document, "Laurel" and "Columbia" also share instances in the US state of Maryland. Such context may not be present in the very short documents found on Twitter. Some context may be available across a user's tweet history, but such data may not be available. Other techniques, then, are needed for this data stream.

In this paper, we demonstrate a new technique called RapidGeo for extracting and disambiguating place names from a location specific Twitter feed (tweets from Tuscaloosa following the April 27, 2011 tornado), using an unsupervised technique for tagging location mentions and relying on the known geographic context of a given Twitter feed for disambiguation. In the following sections we will discuss related work, our proposed method, empirical analysis, and our conclusions.

RELATED WORK

While the development of named entity recognizers (NER) is an established part of natural language processing, little work has been done in relation to social media sites like Twitter. Finin, *et al.* (2010) describe work on crowd-sourcing the annotation of tweets using Amazon Mechanical Turk and Crowd Flower. Locke and Martin (2009) describe the annotation of entities in tweets and the training of a *support vector machine* (SVM) from that data. They reported an F-score of 54% on location mentions and, based on this poor performance, suggest transfer learning as an approach in order to leverage labeled corpora from more traditional sources.

Toponym disambiguation has been extensively studied. Techniques range from simple heuristics (Fink *et al.* 2009) to techniques that leverage the information captured in the contextual information present across toponym mentions in a given document (Smith and Crane 2001; Leidner, Sinclair, and Webber 2003; Amitay *et al.* 2004; Zong *et al.* 2005). Another approach was described by Overell and Roger (2008) who crawled Wikipedia articles and built co-occurrence models for place names that can be used for disambiguation location entity mentions in text. This is a relatively mature area of study, though most approaches rely on contextual clues in the language surrounding a mention; context that is likely to not be present in a tweet.

METHOD

A *gazetteer* is simply a dictionary that maps toponyms to geographic coordinates. Formally, let

$$G = \{\langle a_1, \ell_1 \rangle, \langle a_2, \ell_2 \rangle, \ldots\}$$

be a gazetteer consisting of a set of pairs of toponyms, a_i, and associated locations, ℓ_i. Given G and a query string s produced at reference location ℓ_s, this section describes an efficient algorithm for labeling substrings within s that likely match a toponym in G. For example, if

$$G = \{ \quad \langle \text{“District of Columbia”, the latitude and longitude of Washington, D.C., USA} \rangle, \\ \langle \text{“Columbia”, the latitude and longitude of the city of Columbia, Maryland, USA} \rangle, \\ \langle \text{“Colombia”, the latitude and longitude of the Republic of Colombia} \rangle \quad \},$$

Note the difference in spelling.

and the input query is

$$\begin{aligned} s &= \text{“I am going to Columbia”} \\ \ell_s &= \text{the latitude and longitude of a location within the Republic of Panamá,} \end{aligned}$$

our method will label the substring “Columbia” in s with the gazetteer entry for “Colombia”.

Our method, RapidGeo, is empowered by a data structure that can efficiently perform a “fuzzy” matching between strings, mapping toponyms to likely locations in the gazetteer. The fuzziness is what allows for us to mitigate misspellings in the input data, *e.g.*, “Columbia” *vs.* “Colombia.” This is achieved by hashing the gazetteer's toponyms based on a phonetic encoding algorithm—in our case, Double Metaphone (Phillips 2000)—which roughly encodes the way a word would be pronounced by an Anglophone. Therefore, heterographs[1] like “Columbia” and “Colombia” are hashed the to the same value. Our data structure maps each distinct phonetic encoding of the toponyms in G to a K-D Tree data structure (Bentley 1975) containing the associated toponyms in G indexed by their location values. The initialization of this data structure is given in Algorithm 1. Each K-D Tree can be constructed in $O(\kappa \log \kappa)$ time, where κ is the maximum the number of homophones in the gazetteer. The K-D Tree allows us to efficiently perform a nearest neighbor search, *i.e.*, given a location ℓ we can efficiently find the entry in G with the location closest to ℓ. Nearest neighbor search using this method has an empirically observed average case running time of $O(\log \kappa)$ (Bentley 1975). The method for performing the fuzzy matching on a query word is given in Algorithm 2.

Algorithm 1 Initialization of the data structures.

procedure INITIALIZE-INDEX(G)
Require: G is a gazetteer consisting of a set of pairs of names and associated locations.
Ensure: The global data structure R is generated.
    $R \leftarrow$ an empty hashtable mapping phonetic encodings to K-D Tree data structures
    for all $\langle a_i, \ell_i \rangle \in G$ **do**
        $h \leftarrow$ PHONETICALLY-ENCODE(a_i)
        if $h \notin R$ **then**
            set $R(h) \mapsto$ an empty K-D Tree
        end if
        add a_i to $R(h)$ indexed by ℓ_i
    end for
end procedure

Next, the input string s is tokenized into a set of substrings, $S = \{s_1, s_2, \ldots\}$, by splitting it at non-word characters. The set of all possible n-word-gram sub-sequences of tokens is then enumerated in order of decreasing size, as defined in Algorithm 3. For example, the input string “I am going to Columbia” would be tokenized to $S = \{$“I”, “am”, “going”, “to”, “Columbia”$\}$, and POTENTIAL-LOCATION-ITERATOR(S) of Algorithm 3 will produce the following enumeration of substrings: {“I am going to Columbia”, “I am going to”, “am going to Columbia”, “I am going”, “am going to”, “going to Columbia”, “I am”, “am going”, “going to”, “to Columbia”, “I”, “am”, “going”, “to”, “Columbia”}. Each of these substrings can then be matched against the data structure using Algorithm 2. One should note that this enumeration will produce $\frac{|S|}{2}(|S|+1)$ substrings; in Twitter, for example, the character limit restricts $|S|$ to at most 70, which results in a very tractable maximum of 2485 substrings that need to be matched. In most cases this number will even be much smaller.

[1] homophones that have different spellings and meanings.

Algorithm 2 Match a query string against all toponyms in G.

procedure MATCH-LOCATION(s, ℓ_s)
Require: s is a single word to be matched and ℓ_s is a reference location.
Ensure: The location in G most likely matching s will be returned.
    $h \leftarrow$ PHONETICALLY-ENCODE(s)
    if $h \notin R$ **then**
        return $\varnothing$
    else
        return NEAREST-NEIGHBOR($R(h), \ell_s$) /* find the homophone to s in G that has location closest to ℓ_s */
    end if
end procedure

Algorithm 3 Enumerating potential toponym substrings to match from a query string.

procedure POTENTIAL-LOCATION-ITERATOR(T)
Require: T is a list of tokens, each being a pair $\langle a_i, s_i \rangle$ consisting of an offset a_i and a substring s_i.
Ensure: This iterator will yield all combinations of sequential tokens in T from largest to smallest.
    $i \leftarrow 0$
    $m \leftarrow |T|$
    while $m > 0$ **do**
        $e \leftarrow$ " "
        for $j \leftarrow i$ **to** $i + m - 1$ **do**
            $\langle a_j, s_j \rangle \leftarrow T[j]$
            if $j > i$ **then**
                $e \leftarrow e +$ " "
            end if
            $e \leftarrow e + s_j$
        end for
        yield $\langle i, i + m - 1, e \rangle$
        $i \leftarrow i + 1$
        if $i + m > |T|$ **then**
            $i = 0$
            $m \leftarrow m - 1$
        end if
    end while
end procedure

There is also the matter of nested toponyms. For example, we do not want the query string "District of Columbia" to be labeled twice:

$$\text{“District of } \underbrace{\underbrace{\text{Columbia}}_{\text{Columbia, Maryland, USA}}}_{\text{Washington, D.C., USA}}\text{”.}$$

Therefore, we need a way of preventing nested matches from being re-labeled. This is accomplished by maintaining a hash of which portions of the query string have already been labeled and avoiding enumerating/matching any future substrings in that region.

Finally, we want to limit highly unlikely labels. For example, the gazetteer might contain a toponym for the Tyrolean municipality of "Going" in Austria, however, we would not want the input query of "I am going to Columbia" to label the substring "going" with that toponym unless the reference location is relatively close to Austria. Likewise, the gazetteer might have an entry for the country of Armenia under its ISO country code "AM", and we would not want the word "am" to match that toponym. Therefore, we provide a threshold, γ, that defines the maximum allowable distance between the query's reference location and a potential toponym match's location.

This entire process is implemented in Algorithm 4. Assuming a hashtable with constant time access and addition is utilized, initialization of the data structures (Algorithm 1) will run in worst case $O\left(\frac{|G|}{\nu}\kappa\log\kappa\right)$ time and each query will run in average case $O(\log\kappa)$ time, where κ and ν are, respectively, the maximum and minimum number of homophones in G.

Algorithm 4 Labels all toponyms in a query string.

1: **procedure** LABEL-LOCATIONS(s, ℓ_s, γ)
Require: $||p-q||_d$ denotes a distance metric between locations p and q. γ is a threshold for the maximum distance between ℓ_s and the labeled toponym.
2: $T \leftarrow$ TOKENIZE(s)
3: $M \leftarrow$ an array of $|T|$ booleans all initialized to TRUE
4: **for all** $\langle i, j, s_i\rangle \in$ POTENTIAL-LOCATION-ITERATOR(T) **do**
5: **if** $\left(\bigwedge_{k=i}^{j} M[k]\right) \wedge (\langle a_m, \ell_m\rangle \leftarrow \text{MATCH-LOCATION}(s_i, \ell_s)) \neq \varnothing \wedge ||\ell_s - \ell_m||_d \leqslant \gamma$ **then**
6: label tokens $T[i]$ through $T[j]$ as matching gazetteer entry $\langle a_m, \ell_m\rangle$
7: **for** $k \leftarrow i$ **to** j **do**
8: $M[k] \leftarrow$ FALSE
9: **end for**
10: **end if**
11: **end for**
12: **end procedure**

EMPIRICAL ANALYSIS

The work in this paper is based on Tweets collected from nearby Tuscaloosa, Alabama between April 28 and May 9, 2011. We used Twitter's Search API's geocode method to query for tweets by location, using the center coordinates of Tuscaloosa with a radius of 15 miles. For users whose tweets were captured using this method, we subsequently used the Search API's search method to obtain any of their other tweets that might not have been made in that region.

The result returned for each tweet via the geocode method contains metadata about user location, language and the client application used when posting the tweet. Location information is available from two sources: a field containing the location from the user's profile and a field containing a latitude/longitude pair populated using the optional geotagging feature supported by mobile Twitter clients (users must opt-in to use this feature). We use these fields to calculate a user's location at the time of the tweet. If the profile location is a latitude/longitude pair (either from a mobile client updating the profile location or from the geotagging feature) we use the `geonames.org` web service to find all locations within 25 miles of the coordinates and choose the closest populated place. If the location is given as a string, `geonames.org` is queried with the string for a match. The geographic information returned by `geonames.org` includes a place name, second administrative level (state or province), country code, latitude and longitude. For our

work, we used the geotagging location, if present. Otherwise, user location was based on the profile location. When applying our algorithm, the most recent location assigned to a tweet's author is used as the reference location.

We used a MySQL serialization of the `geonames.org` gazetteer to build our place name index and associated K-D trees. Each toponym in the database has a canonical name (for example, "Tuscaloosa") and a set of alternate names, and we generate index entries for all possible Latin script names for a location, associating with each the latitude and longitude and administrative level information (*i.e.*, state and county).

To validate RapidGeo, we annotated 500 tweets selected at random from the tweets we collected between the dates of April 28 and May 5, 2011. We asked a single annotator to mark spans of text that referred to a location. Of the 500 tweets, 73 had one or more location mentions for a total of 99 mentions. We next manually assigned a toponym from the `geonames.org` gazetteer to each mention, resulting in set of 69 validation instances. For example, the mention of "Pleasant Grove" in the tweet "another arrest in Pleasant Grove for Looting [photo url]" was assigned to Pleasant Grove, Alabama. For the experiments discussed below, tweets were processed as is, with no normalization or clean-up of the text.

We first wanted a baseline for the performance of an off-the-shelf named entity recognizer. We applied the Stanford NER[2] against the 500 annotated tweets, using the 69 labeled and disambiguated mentions as our ground truth annotations.

We next tested our algorithm on the same data. The distance threshold parameter, γ, was varied exponentially according to the function $\gamma(i) \mapsto 1.085^i$ for $i \in \{0,\ldots,99\} \cup \{\infty\}$, resulting in 100 values for γ ranging from 1 to 2966 kilometers (and also ∞).

We compare the location name recognition and labeling portion of RapidGeo against that of the Stanford NER using two metrics: exact matching (*i.e.*, the tagged spans match exactly to the annotated ground truth) and overlap (*i.e.*, the tagged spans overlap with a portion of the annotated ground truth). These results are given in Figures 1 and 2, respectively. For each metric we calculate the accuracy (the ratio of true positives to the sum of true positives, false positives, and false negatives) precision (the percentage of all labels that agreed with ground truth) and recall (the ratio of true positives to the sum of true positives and false negatives) of both the NER and RapidGeo. Note that, while the NER is also capable of identifying other types of named entities, we only used what the NER tagged as locations or organizations. In both cases the NER performed poorly, with low accuracy and precision, but somewhat higher recall. This supports the intuition that an standard NER would not do well on this text genre.

Finally, in order to evaluate the precision of RapidGeo, we compare the tagged latitude and longitude of correctly labeled toponyms against the associated geographic coordinates from the ground truth, using great circle distance (in kilometers) as the metric. These data are presented in Figure 3. As we can see, median error is close to 0km until about $\gamma = 200$km, at which point the error spikes to about 75km. One potential explanation for this error is that RapidGeo will always prefer matches that are geographically closer to the reference location. For example, the gazetteer has an entry for both the City of Tuscaloosa and the County of Tuscaloosa, and both are indexed under the alternate simple name of "Tuscaloosa". Each toponym, however, has a different geographic coordinate, so while the human annotator might arbitrarily decide to label all of the locations as the City of Tuscaloosa in the ground truth, RapidGeo will decide between the city and the county based off of proximity to the reference location. These discrepancies are what contribute to the error. As the distance threshold γ is increased, more potentially incorrect alternative locations are considered, which is what further increases the error. As another example, many tweets mentioned "The White House". While the human obviously correctly geotagged those mentions as the building in Washington, D.C., our gazetteer also included another historic monument called "The White House" located near Helena, Arkansas. Since Arkansas is closer to the reference locations of most of the tweets (*i.e.*, places in Alabama) than Washington, D.C., some of the mentions were incorrectly tagged by RapidGeo as being in Arkansas when γ is large.

Additional examples of RapidGeo's tagging and disambiguation are provided in Table 1.

[2] `http://nlp.stanford.edu/software/CRF-NER.shtml`

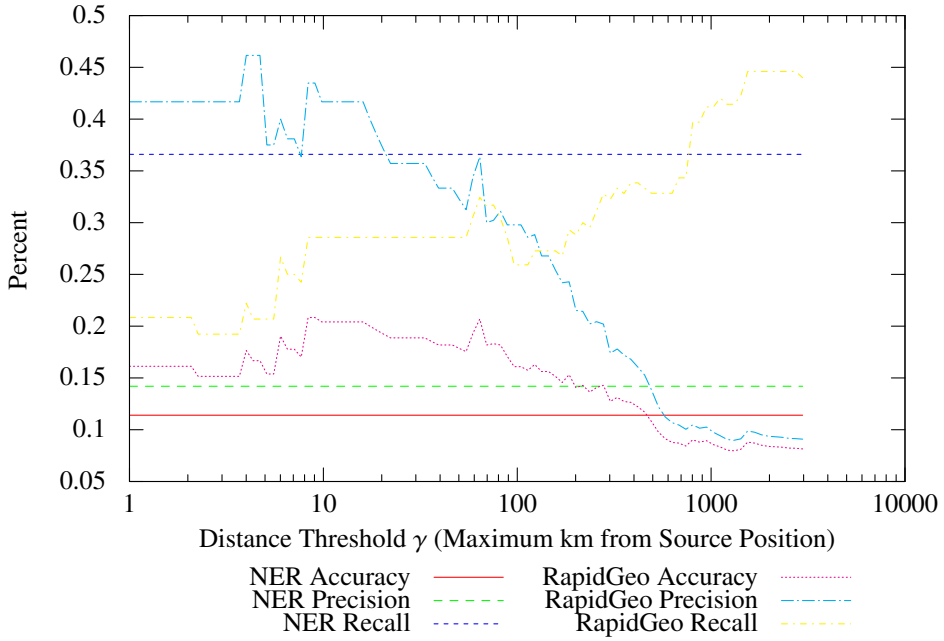

Figure 1: Performance of RapidGeo using the exact match metric for varying distance thresholds.

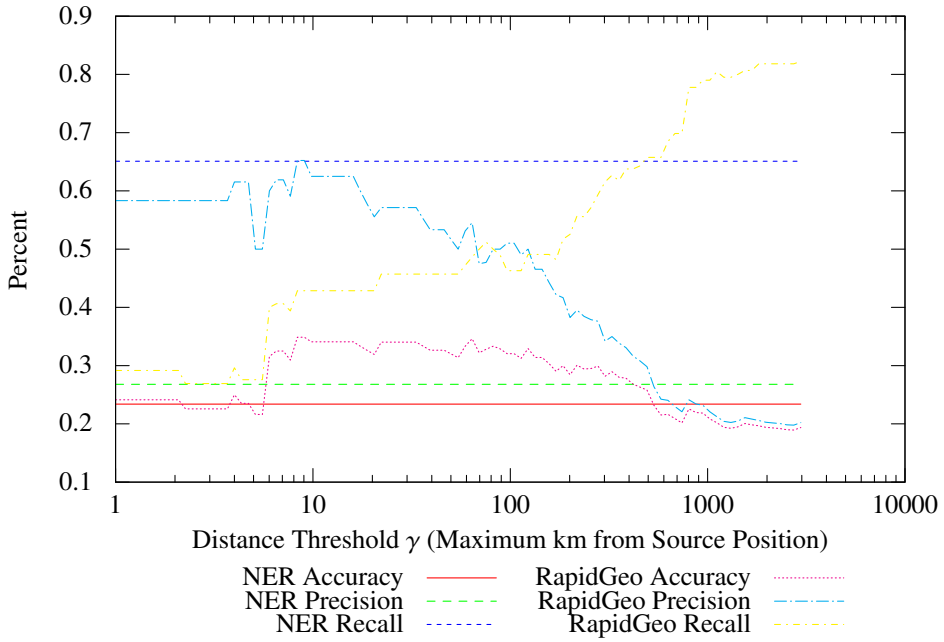

Figure 2: Performance of RapidGeo using the overlap metric for varying distance thresholds.

TWEET (s)	TWEET LOCATION (ℓ_s)	RAPIDGEO TAGS
RT @USER: "@USER: @USER we are at Pratt city, just found two bodies down the street, so afraid many more to find...." #WeAreAlabama	Birmingham, AL (33°31′14.38″N, 87°11′51.04″W)	"Pratt city" ↦ Pratt City, AL (33°32′24.40″N, 87°7′52.03″W)
MT @USER We need volunteers immediately. // Address: 3600 Third Ave South, Birmingham, AL 35222	Tuscaloosa, AL (33°12′35.43″N, 88°25′50.98″W)	"Birmingham" ↦ Birmingham, AL (33°31′14.38″N, 87°11′51.04″W)
RT @USER: RT @USER: 5pts Baptist, Northport, AL NEEDS Boxes, packing tape & storage bins NOW!! Please! URGENT! #WeAreAlabama	Tuscaloosa, AL (33°12′35.43″N, 88°25′50.98″W)	"Northport" ↦ City of Northport, AL (33°15′27.18″N, 88°24′6.26″W)
Volunteers also needed at Holt High School.	Tuscaloosa, AL (33°12′35.43″N, 88°25′50.98″W)	"Hold High School" ↦ Holt High School (33°14′12.41″N, 88°30′45.00″W)
RT @USER: First Wesleyan Church is currently at capacity for housing First Responders at our location until further notice. Thanks.	Tuscaloosa, AL (33°12′35.43″N, 88°25′50.98″W)	"First Wesleyan Church" ↦ First Wesleyan Church (33°13′53.44″N, 88°27′53.96″W)
:: Holy Spirit is accepting volunteers. They are in need of Spanish translators, too.	Tuscaloosa, AL (33°12′35.43″N, 88°25′50.98″W)	"Holy Spirit" ↦ Holy Spirit School (34°41′26.34″N, 87°25′37.02″W)
another arrest in Pleasant Grove for Looting `http://yfrog.com/h6dazgtj`	Tuscaloosa (33°12′35.43″N, 88°25′50.98″W)	"Pleasant Grove" ↦ City of Pleasant Grove, AL (33°29′36.82″N, 87°1′20.00″W)
RT @USER: Calker Co. EMA in need of baby formula, wipes, diapers. donations can be dropped off at the EMA Building in Jasper #WeAreAlabama	Tuscaloosa (33°12′35.43″N, 88°25′50.98″W)	"Jasper" ↦ Jasper, AL (33°49′52.39″N, 88°43′20.96″W)

Table 1: Examples of RapidGeo's tagging an disambiguation of the Tuscaloosa dataset. Twitter usernames have been redacted to maintain privacy.

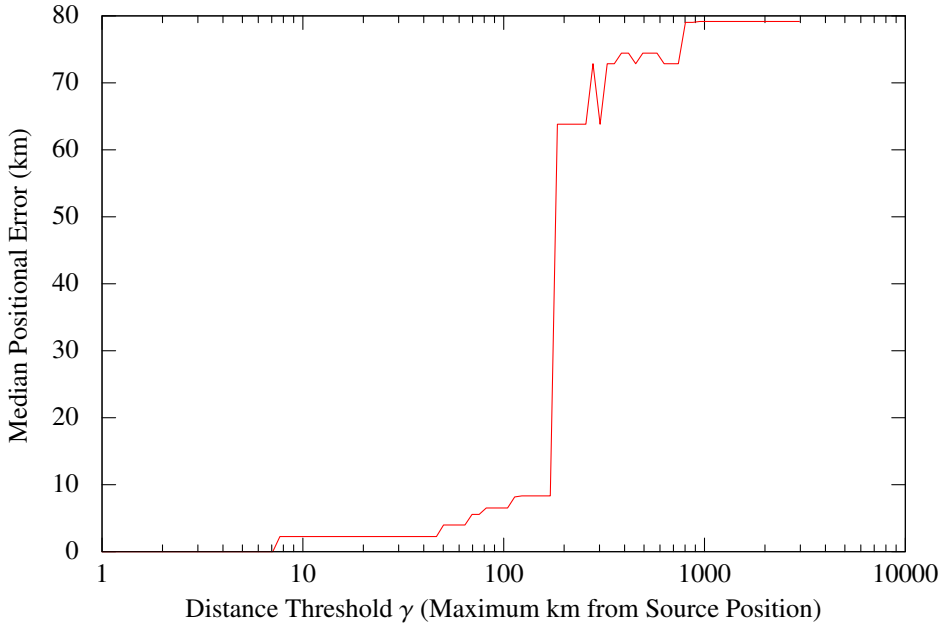

Figure 3: Precision of RapidGeo: Median error in the geographic coordinates assigned to tags.

CONCLUSIONS

Part of the reason why RapidGeo's false positive rate increases with proportion to γ is that it currently does not analyze the semantic context of the queries. For example, the Tuscaloosa tornadoes were concurrent with the killing of Osama bin Laden, which was thereby the topic of many tweets. Unfortunately, there is a place in Kentucky called "Laden"; when γ was set large enough to include Kentucky, all of the tweets referencing Osama bin Laden were geotagged to Laden, Kentucky. One potential way of mitigating problems like these in the future would be to create a hybrid approach that fuses the output of an NER with RapidGeo. For example, a properly trained NER could identify the noun "Osama bin Laden" as being that of a person's name, allowing RapidGeo to ignore it.

Some of RapidGeo's positional error is also due to incorrectly identified misspellings. Experimenting with phonetic encoding algorithms other than Double Metaphone to correct for misspellings is planned for future work.

In this paper we have demonstrated a novel method for the unsupervised extraction of location references from tweets. Such techniques are required for such data sources given the noisy characteristics and informality of the text and shifting topical domains. The results we report, as expected, beat an off-the-shelf NER, though additional work needs to be done to improve these numbers. Additional, high reliability validation corpora also need to be developed for Twitter if methods such as what we have described are to progress.

REFERENCES

1. Amitay, E.; Har'El, N.; Sivan, R.; and Soffer, A. 2004. Web-a-where: geotagging web content. In *Proceedings of the 27th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, 273–280. ACM.

2. Bentley, J. L. 1975. Multidimensional binary search trees used for associative searching. *Communications of the ACM* 18:509–517.

3. Finin, T.; Murnane, W.; Karandikar, A.; Keller, N.; Martineau, J.; and Dredze, M. 2010. Annotating Named Entities in Twitter Data with Crowdsourcing. *Computational Linguistics* 2010(June):80–88.

4. Fink, C.; Piatko, C.; Mayfield, J.; Chou, D.; Finin, T.; and Martineau, J. 2009. The geolocation of web logs from textual clues. In *Proceedings of the International Conference on Computational Science and Engineering*, volume 4, 1088–1092. IEEE.

5. Hughes, A. L., and Palen, L. 2009. Twitter adoption and use in mass convergence and emergency events. *International Journal of Emergency Management* 6(3/4):248.

6. Leidner, J. L.; Sinclair, G.; and Webber, B. 2003. Grounding spatial named entities for information extraction and question answering. *Proceedings of the HLT-NAACL Workshop on Analysis of Geographic References* 1:31–38.

7. Locke, B. 2009. Named entity recognition: Adapting to microblogging. University of Colorado Boulder Senior Thesis. Advisor: James Martin.

8. Mendoza, M., and Poblete, B. 2010. Twitter under crisis: Can we trust what we RT? *Proceedings of the First Workshop on Social Media Analytics*.

9. Overell, S., and Rüger, S. 2008. Using co-occurrence models for placename disambiguation. *International Journal of Geographical Information Science* 22(3):265–287.

10. Phillips, L. 2000. The double metaphone search algorithm. *C/C++ Users Journal*.

11. Smith, D., and Crane, G. 2001. Disambiguating geographic names in a historical digital library. *Research and Advanced Technology for Digital Libraries* 127–136.

12. Starbird, K., and Palen, L. 2010. Pass it on ?: Retweeting in mass emergency. In *Proceedings of the 7th International ISCRAM Conference*.

13. Verma, S.; Vieweg, S.; Corvey, W. J.; Palen, L.; Martin, J. H.; Palmer, M.; Schram, A.; and Anderson, K. M. 2011. Natural language processing to the rescue?: Extracting "situational awareness" tweets during mass emergency. *Artificial Intelligence*.

14. Zong, W.; Wu, D.; Sun, A.; Lim, E.-P.; and Goh, D. H.-L. 2005. On assigning place names to geography related web pages. In *Proceedings of the 5th ACM/IEEE-CS Joint Conference on Digital Libraries*, 354. New York, New York, USA: ACM Press.