When Twitter meets Foursquare: Tweet Location Prediction using Foursquare

Kisung Lee †, Raghu K. Ganti ¶, Mudhakar Srivatsa ¶, Ling Liu †
†College of Computing, Georgia Institute of Technology, Atlanta, GA USA
¶IBM T. J. Watson Research Center, Yorktown Heights, NY USA
kslee@gatech.edu, {rganti, msrivats}@us.ibm.com, lingliu@cc.gatech.edu

ABSTRACT

The continued explosion of Twitter data has opened doors for many applications, such as location-based advertisement and entertainment using smartphones. Unfortunately, only about 0.58 percent of tweets are geo-tagged to date. To tackle the location sparseness problem, this paper presents a methodical approach to increasing the number of geo-tagged tweets by predicting the fine-grained location of those tweets in which their location can be inferred with high confidence. In order to predict the fine-grained location of tweets, we first build probabilistic models for locations using unstructured short messages tightly coupled with semantic locations. Based on the probabilistic models, we propose a 3-step technique (Filtering-Ranking-Validating) for tweet location prediction. In the filtering step, we introduce text analysis techniques to filter out those location-neutral tweets, which may not be related to any location at all. In the ranking step, we utilize ranking techniques to select the best candidate location for a tweet. Finally, in the validating step, we develop a classification-based prediction validation method to verify the location of where the tweet was actually written. We conduct extensive experiments using tweets covering three months and the results show that our approach can increase the number of geo-tagged tweets 4.8 times compared to the original Twitter data and place 34% of predicted tweets within 250m from their actual location.

1. INTRODUCTION

With the continued advances of social network services, such as Twitter, Facebook and Foursquare, a tremendous amount of unstructured textual data has been generated. One of the most popular forms of such unstructured texts is a short text message, called *tweet*, from Twitter and each tweet has up to 140 characters. Twitter users are posting tweets about almost everything from daily routine, breaking news, score updates of various sport events to political opinions and flashmobs [16, 27]. Over hundreds of millions of such tweets are generated daily. Furthermore, more and more business organizations recognize the importance of Twitter and provide their customer services through Twitter, such as receiving feedback about products and responding to customers' questions using tweets [5].

Tweets can be much more valuable when tagged with their location information because such geo-tagged tweets can open new opportunities for many applications. For example, if a user posts a tweet tagged with her current location, nearby local stores can immediately send her customized coupons based on the context of the tweet or her profile assuming that she is a subscriber of such location-based advertisement services. Similarly, local news and places of interest can be recommended based on the location, the context of the tweet and the past experiences of her friends in a social network. Geo-tagged tweets can also be used to report or detect unexpected events, such as earthquakes[24], robbery or gun shots, and notify the event to the right people instantly, including those who are close to the location of the event.

On one hand, like most social network services, Twitter recognizes the value of tagging tweets with location information and provides the geo-tagging feature to all its users. On the other hand, such opt-in geo-tagging feature is confronted with several challenges. First, Twitter users have been lukewarm in terms of adopting the geo-tagging feature. According to our recent statistical analysis over 1 billion tweets spanning three months, only 0.58% tweets have their fine-grained location. With such a tiny amount of geo-tagged tweets, it would be very hard to realize the many social and business opportunities such as those mentioned above. Second, even for the limited tweets tagged with geometric coordinates, a fair amount of them cannot be used effectively because their geometric coordinates cannot be served as quality indicators of useful semantic locations, such as points of interest and places where events of interest may happen or have happened. This location sparseness problem makes it very challenging for identifying the types of tweets in which we can infer their location information, i.e., the location where a tweet was written. We argue that in order to derive new values and insights from the huge amount of tweets generated daily by Twitter users and to better serve them with many location-based services, it is important to have more geo-tagged tweets with semantically meaningful locations.

In this paper we present a methodical approach to increasing the number of geo-tagged tweets by predicting the fine-grained location of *each tweet* using a multi-source and multi-model based inference framework. Our focus is to pre-

WWW '14 Seoul, Korea

dict the location of carefully selected tweets in which their location can be inferred with high confidence based only on their textual data, instead of trying to predict the location of all (or most) tweets. First of all, we address the location sparseness problem of Twitter by building the probabilistic models for locations using unstructured short messages that are tightly coupled with their semantic locations. In order to achieve the tight coupling between text and location, we propose to use Foursquare - a popular location-centric social network, as a source for building these probabilistic models. Based on the probabilistic models, we propose a 3-step technique (Filtering-Ranking-Validating) for predicting the fine-grained location of tweets. In the *filtering* step, we develop a set of filters that can remove those location-neutral tweets, which may not be related to any location at all, prior to entering the location prediction (ranking) phase. This effort enables us to filter out as many location-neutral tweets as possible to minimize the noise level and improve the accuracy of our location prediction model. In the *ranking* step, candidate locations for each tweet are determined using one of the three ranking techniques: standard machine learning approaches, naive Bayes model and $tfidf$ value. Once the top ranked location is assigned to the tweet, in the *validating* step, we utilize a classification-based prediction validation method to accurately predict the location where the tweet was actually written. We report our experimental evaluation conducted using a set of tweets, collected over a three-month period in New York City. The results show that our approach can increase the number of geo-tagged tweets 4.8 times compared to the original Twitter data and place 34% of predicted tweets within 250m from their actual location.

2. RELATED WORK

We categorize the related work into four categories: 1) location prediction in Twitter-like social networks, 2) topic and user group prediction in Twitter-like social networks, 3) analysis of Foursquare check-ins, and 4) location prediction using other online contents.

Location prediction in social networks. Existing work can be divided into the problem of predicting the location of *each Twitter user* [11, 13, 19] or predicting the location of *each tweet* [14, 17]. Concretely, [11] proposes a technique to predict the *city-level* location of each Twitter user. It builds a probability model for each city using tweets of those users located in the city. Then it estimates the probability of a new user being located in a city using the city's probability model and assigns the city with the highest probability as the city of this new user. To increase the accuracy of the location prediction, it utilizes local words and applies some smoothing techniques. [13] uses a Multinomial Naive Bayes model to predict the *country* and *state* of each Twitter user. It also utilizes selected region-specific terms to increase the prediction accuracy. [19] presents an algorithm for predicting the home location of Twitter users. It builds a set of different classifiers, such as statistical classifiers using words, hashtags or place names of tweets and heuristics classifiers using the frequency of place names or Foursquare check-ins, and then creates an ensemble of the classifiers to improve the prediction accuracy. These coarse-grained location prediction methods rely heavily on the availability of a large training set. For example, the number of tweets from the users in the same city can be quite large and comprehensive. In contrast, the goal of our work is to predict the fine-grained location of each tweet if the tweet can be inferred with high confidence.

[14] and [17] are the most relevant existing work as they centered on predicting the location of each tweet. [17] builds a POI (Place of Interest) model, assuming that a set of POIs are given, using a set of tweets and web pages returned by a search engine. For a query tweet, it generates a language model of the tweet and then compares it with the model of each POI using the KL divergence to rank POIs. Since it uses only 10 POIs and a small test set for its evaluation, it is unclear how effective the approach is in a real-world environment in which there are many POIs and a huge number of tweets and furthermore many tweets contain noisy text, irrelevant to any POI. [14] extracts a set of keywords for each location using tweets from location-sharing services, such as Foursquare check-in tweets, and other general expression tweets posted during a similar time frame. To predict the location of a new tweet, it generates a keyword list of the tweet and compares it with the extracted keywords of locations using cosine similarity. An obvious problem with this work is that it treats all tweets equally in the context of location prediction. Thus, it suffers from high error rate in the prediction results, especially for those location-neutral tweets.

Topic and user group prediction in social networks. In addition to location prediction of Twitter data, other research efforts have been engaged in inferring other types of information from Twitter data. [18] proposes a framework to predict topics of each tweet. It builds a language model for each topic using hashtags of tweets and evaluates various smoothing techniques. [23] proposes a social network user classification approach, which consists of a machine learning algorithm and a graph-based label updating function. [8] proposes an approach to predict sentiments of tweets and [9] presents a technique to classify Twitter users as either spammers or nonspammers. Most of the work in this category build their language-based classification model using supervised learning and utilize some external knowledge to initialize the classification rules, such as spam or non-spam. In contrast to this line of work, we focus on location detection of tweets rather than Twitter user classification.

Analysis of Foursquare check-ins. [12, 22] analyze Foursquare check-in history in various aspects. [12] shows spatial and temporal (daily and weekly) distribution of Foursquare check-ins. It also analyzes the spatial coverage of each user and its relationship with city population, average household income, etc. [22] also shows spatio-temporal patterns of Foursquare check-ins and calculates the transition probabilities among location categories.

Location prediction using other online contents. Many studies have been conducted to infer the geographical origin of online contents such as photos [26], webpages [7] and web search query logs [15]. [26] builds a language model for each location (a grid cell) using the terms people use to describe images. [7] identifies geographical terms in webpages using a gazetteer to infer a geographical focus for the entire page. [15] utilizes a geo-parsing software which returns a list of locations for web search query logs to infer the location of users (at zip code level).

3. OVERVIEW

In this section we first describe the reference data models

for Twitter and Foursquare data. Then we describe how we build the language models for locations of tweets, using short text messages of Foursquare. Finally we outline the design principles and the system architecture of our location prediction framework.

3.1 Twitter Reference Model

Twitter is the most representative microblogging service being used widely, from breaking news, live sports score updates, chats with friends (called *followers*) to advertising and customer service by many companies. Twitter data consists of tweets. Formally, a tweet is defined by a user ID, a timestamp when the tweet was posted, and a short text message up to 140 characters. To enrich its data with location information, Twitter provides not only a location field for each user but also a feature for geo-tagging *each tweet* [2]. Therefore each tweet can be tagged with a fine-grained location, such as a geometric coordinate defined by a latitude and longitude, though the number of tweets with the geo-tag is very small. Our prediction framework performs the location prediction solely based on the short unstructured text messages without requiring user ID and timestamp of tweets. In order to perform text analysis over all tweets, we formally model each tweet as a vector of words in our word vocabulary of n words, denoted by $< w_1, w_2, \ldots, w_n >$. For each tweet tx, if w_1 appears 2 times in tx, we have a value 2 in the position of w_1. Thus, a tweet vector is a vector of n elements of integer type with each element tx_i $(1 \leq i \leq n)$ denoting the number of occurrences of the word w_i in tx. To get a list of words from tweets, we process each tweet by breaking the tweet into tokens, stemming the tokens, and removing stop words from them.

3.2 Foursquare Reference Model

Foursquare is a social network service, which is specialized in location-sharing through check-ins. As of September 2013 [1], there are over 40 million users and over 4.5 billion check-ins, with millions more every day. Users can check into a place by selecting one of the nearby places from their current location (usually using their smartphones with GPS), and leave tips for a specific place. Each tip has up to 200 characters and is explicitly associated with one place. Foursquare provides the basic information of places, such as name, address, website URL, latitude and longitude, and category. A fair number of Foursquare users are linking their Foursquare account with their Twitter account such that their check-ins are automatically posted to their Twitter account. We argue that building probabilistic language models for locations using Foursquare tips will be the first step towards developing a methodical approach to high quality location prediction for each tweet. Concretely, in order to integrate Foursquare as an external location-specific data source for predicting the location of each tweet, we formally model each tip in Foursquare based on our Twitter vocabulary of n words. Thus, a tip *tip* is also represented as a vector of n elements of integer type, with each element tip_i denoting the number of occurrences of the word w_i in *tip*. Each tip is also associated with a location l. Similar to tweet tokenization process, we get a list of words from tips by breaking each Foursquare tip into tokens, stemming the tokens, and removing stop words from them.

3.3 Location Modeling

In contrast to many existing approaches [11, 13, 19, 17, 14], which mainly use geo-tagged tweets to build a probabilistic model for each location, we argue that a high quality location model for tweets should identify those geometric coordinates that are actually associated with some semantically meaningful place(s) of interest (PoI) and build the location models only for those semantic locations, instead of building a location model for every geometric coordinate captured by some tweets. For example, there are many tweets which are not related to any location at all since people can tweet anything regardless of their location. We refer to those tweets that do not relate to any semantic location at all as *location-neutral* tweets. Clearly, if too many such location-neutral tweets are involved in location modeling, the language models we build for locations can be both noisy and misleading. Alternatively, if we counter the sparseness problem of geo-tagged tweets by dividing the geographical region of interest into multiple partitions (such as grids) and then building a language model using tweets generated in each partition, it will also be misleading since each partition may include tweets from multiple locations and it is hard to differentiate tweets written in one location from those written in another location because each geo-tagged tweet has only latitude and longitude. This problem can be aggravated by the sampling errors existing in most of the localization techniques.

Foursquare, as a location-sharing social network service, has a collection of PoIs (places of interest), and each tip is associated with a short text message and a PoI. This makes Foursquare a valuable resource for building good probabilistic language models for locations, because Foursquare data includes one of the best human-encoded mappings of geometric locations to semantic locations (PoIs) as well as a set of short messages (tips) for them. This motivates us to use Foursquare tips instead of noisy tweets to build more accurate and dependable probabilistic models for locations. In the situation where multiple locations have the same latitude and longitude (such as multistory buildings), we can build a separate language model for each location based on the corresponding PoIs and the set of tips associated with the PoIs.

Let the set of locations (PoIs) in Foursquare be $l_1, l_2, \ldots, l_m$. To predict the location of tweets using the probabilistic models of locations, we first build a language model (LM) for each Foursquare location using a set of tips associated to that location. The language model has a probability for each word (unigram model) or each sequence of n words (n-gram model). Let $tf(w,t)$ denote the number of occurrence of word w in the tip t, $c(w,l)$ denote the number of occurrences of word w in all tips associated to location l and n be the number of all words in our word vocabulary. We calculate the probability of a word w in a location l using the frequency-based maximum likelihood estimation as follows:

$$p(w,l) = \frac{c(w,l)}{\sum_{i=1}^{n} c(w_i,l)}, \quad c(w,l) = \sum_{tip \in tips(l)} tf(w,tip)$$

where $tips(l)$ is the set of tips associated to location l. Given that there are some Foursquare locations with a very small number of associated tips, in order to generate dependable LMs using a sufficient number of tips, we build LMs only for locations with more than a minimum number of tips, defined

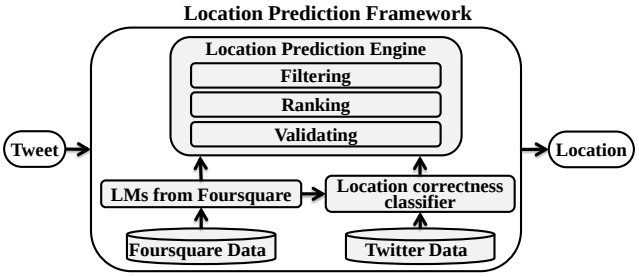


Figure 1: Framework Architecture

by a system-supplied parameter θ_{tip} and also consider only commonly used words in modeling each location.

Bigram Language Model. Instead of the unigram models, where the language model has a probability for each word, we can define a probability for each sequence of n words (n-gram model). For presentation brevity, we below present a bigram model, which can be easily extended to n-gram models. Let $p(w_{i-1}w_i, l)$ be the probability of a bigram $w_{i-1}w_i$ in the tips of location l. The probability of a location l for a tweet T using the bigram LMs is computed as follows:

$$p(l \mid T) = \prod_{w_{i-1}w_i \in T} p(w_{i-1}w_i, l)$$

To estimate the probability of bigrams by handling unobserved bigrams in the tips, in this paper, we explore three different smoothing techniques: Laplace smoothing, Absolute discounting and Jelinek-Mercer smoothing [10]. The three smoothing techniques are defined as follows:

Laplace smoothing, which adds 1 to the frequency count of each bigram. This is defined as follows, where $c(w_{i-1}w_i, l)$ is the frequency count of a bigram $w_{i-1}w_i$ included in the tips of location l:

$$p(w_{i-1}w_i, l) = \frac{1 + c(w_{i-1}w_i, l)}{\sum_{w_i}(1 + c(w_{i-1}w_i, l))}$$

Absolute Discounting, which includes interpolation of bigram and unigram LMs by subtracting a fixed discount D from each observed bigram. This is defined as follows, where $N_{w_{i-1}}$ is the number of observed bigrams which starts with w_{i-1} such that $\mid \{w_i : c(w_{i-1}w_i, l) > 0\} \mid$:

$$p(w_{i-1}w_i, l) = \frac{max\{c(w_{i-1}w_i, l) - D, 0\}}{\sum_{w_i} c(w_{i-1}w_i, l)}$$

$$+ \frac{D \cdot N_{w_{i-1}}}{\sum_{w_i} c(w_{i-1}w_i, l)} \cdot \frac{c(w_i, l)}{\sum_{w_i} c(w_i, l)}$$

Jelinek-Mercer smoothing, which linearly interpolates between bigram and unigram LMs using parameter λ:

$$p(w_{i-1}w_i, l) = \lambda \cdot \frac{c(w_{i-1}w_i, l)}{\sum_{w_i} c(w_{i-1}w_i, l)} + (1 - \lambda)\frac{c(w_i, l)}{\sum_{w_i} c(w_i, l)}$$

Intuitively, the unigram LMs might be sufficient for short text messages like tweets. But we will conduct experiments to compare the unigram models with the bigram models in terms of the prediction precision and errors.

3.4 System Architecture

Even though we build dependable language models for locations using Foursquare tips, there are still several unique challenges for prediction of the fine-grained location of each tweet. The first challenge is that there are lots of tweets that may not be related to any location at all. Thus, it is important to distinguish those location-neutral tweets, which are completely irrelevant to any location, from those tweets whose locations can be learned and predicted. For example, some daily mundane tweets, such as "`Have a good day!`", rarely have any hint that can be used to predict their location. To address this we need to develop effective techniques to filter out as many location-neutral tweets as possible to minimize the noise level and improve the accuracy of our location prediction model. The second challenge is that a tweet can refer to another location which is not related to the current location where the tweet was written. For example, it is not unusual that Twitter users post tweets about sports games of their favorite teams even though their current location is not at all related to the locations where the games are being played. Therefore, we also need to develop an approach to detect whether the *referred* location of a tweet, predicted by the location prediction model, is the same as its current location. The referred location of a tweet means the location which is explicitly mentioned or implicitly hinted in the tweet. Finally, to respect the privacy of users, the location prediction model should not depend on user ID and timestamp of the tweets. To address these challenges, we develop a multi-phase location prediction framework that utilizes the probabilistic models of locations built using Foursquare tips.

Figure 1 provides a sketch of our system architecture for predicting the fine-grained location of a tweet. Our location prediction engine consists of three steps: (i) **Filtering**: Identification of "I don't know" tweets, which are also referred to as *location-neutral* tweets, (ii) **Ranking**: Ranking and predicting the referred location of a tweet, which is implied explicitly or implicitly by the text message of the tweet, and (iii) **Validating**: Using the classification model to determine whether there is a match between the *referred location* and the *actual physical location* of that tweet. The filtering step is to identify if a tweet has any location-specific information. Our solution approach uses simple and yet effective pruning techniques to differentiate tweets with location-specific information from tweets having no location-specific hint at all, by utilizing the probabilistic language models for locations built using Foursquare tips (Recall the previous section). This allows us to filter out noisy tweets at early phase of the location prediction process. For those tweets that have passed the filtering step, the ranking step is to select the best matched location among the set of possible locations for each tweet using ranking techniques. Finally, the validating step is to validate whether the predicted location of a tweet is indeed the correct location with respect to the actual location where the tweet was written. We will explain each step in detail in the next section.

4. LOCATION PREDICTION

In this section, we describe the key steps we take to predict the fine-grained location of each tweet and how we utilize the probabilistic language models built based on Foursquare tips and the geo-tagged tweets from Twitter in our location prediction framework. We first discuss how to identify and prune the "I don't know" tweets in the filtering step, and

then we describe how we design the ranking algorithms to select the best location candidate among a set of possibilities for a tweet in the ranking step. Finally, we discuss how to utilize SVM classifier and the geo-tagged tweets as the training data to develop classification models that validate the correctness of the predicted location of a tweet with respect to the actual physical location from where the tweet was generated, in the validating step.

4.1 Filtering Step

We first define "I don't know" tweets as those which have little information about their location or are talking about past or future event. Given a tweet, if there is not any hint about its location, we filter the tweet out because we have no chance of predicting its location using only textual information of the tweet. Also, if a tweet is talking about past or future activities or events, we exclude the tweet because we cannot predict its current actual location even though we may infer the past or future location referred in the tweet. In this paper, the current location of a tweet refers to a location where the tweet was written. To find such "I don't know" tweets, we utilize local keywords and PoS (Part of Speech) tags.

Utilizing local keywords. Even though each Foursquare tip is explicitly coupled with a location, it also includes some words which are too general to represent the location (e.g. "awesome", "menu", "special"). If a tweet consists of only such general words, it would be impossible to predict the tweet's location because many locations have such words and it is hard to differentiate (rank) among the locations. For example, a tweet "`This sun is BLAZING and there's no shade`" has no hint about its fine-grained location because all words in the tweet are too general to represent any location. To extract any hint about fine-grained locations from tweets, we define *local keywords* as a set of words which are representative of a location. To find the local keywords, we calculate the $tfidf$ (Term Frequency, Inverse Document Frequency) [20] score for each word and each location. Let L be the total number of locations and df_w be the number of locations having w in their tips. Our $tfidf$ calculation for a word w and a location l is formally defined as follows:

$$tfidf_{w,l} = p(w,l) \times \log_{10} \frac{L}{df_w}$$

For a word w, if there is any location l in which its score $tfidf_{w,l}$ is larger than a threshold, denoted by θ_{tfidf}, we treat the word w as a local keyword with respect to the location l. If a tweet has no local keyword at all, then we classify the tweet as a "I don't know" tweet. The threshold θ_{tfidf} for choosing local keywords is a tuning parameter in our framework. If we increase the threshold value, a smaller number of local keywords will be selected, and then more tweets could be filtered out as "I don't know" tweets.

Utilizing PoS tags. Even though a tweet has a sufficient number of local keywords, we may not guarantee that the predicted location based on the language models will match the current location with high confidence when the tweet is talking about the future or past event. For example, a tweet "`I'm going to MoMA`" has a local keyword "MoMA" (abbreviation for the Museum of Modern Art in New York City), but is talking about the future location. Therefore, even though we can predict the referred location in the tweet based on the local keywords such as "MoMA" in this example, the predicted location is related to the location where the author of the tweet will be, rather than the current location where this tweet is written. To detect those tweets talking about the past or future location, we utilize PoS (Part-of-Speech) tags generated by a PoS tagger. Given a tweet, if the generated PoS tags of the tweet include any tag about the past tense form, we treat the tweet as a "I don't know" tweet. Since there is no tag about the future tense in existing PoS taggers, we utilize some words related to future or with future sense, such as "will", "going to" and "tomorrow", and remove those tweets that contain such words.

4.2 Ranking Step

After filtering out those location-neutral tweets, we explore three different techniques to rank locations for each of the tweets survived from the filtering step. Given a query tweet, there is a set of candidate locations that are associated to the tweet based on the language models for locations. To predict the location of the tweet, we need to rank all locations and select the location having the highest rank (or top k locations) as the predicted location of the tweet.

Standard Machine Learning Approaches. A most intuitive baseline approach is to build classification models using standard machine learning techniques such as SVM and decision tree. To choose a training set for learning the models, we sample some tips for each location. In our training set, each instance and each feature represent a Foursquare tip and a word respectively. The number of classes in the training set is equal to the number of all locations. Thus, given a tweet, we use the predicted class by the classification models as the predicted location of the tweet.

Naive Bayes Model. Alternatively, given a set of candidate locations for a tweet, we use the simple naive Bayes probabilistic model to rank locations based on the conditional independence assumption among words. Concretely, given a tweet T and the set of possible locations, we calculate the naive Bayes probability for each location l as follows:

$$p(l \mid T) = \frac{p(l) \prod_{w \in T} p(w,l)}{\sum_i p(l_i) \prod_{w \in T} p(w,l_i)}$$

where $p(l)$ is $\frac{1}{L}$ for all locations since in our current implementation we assume the uniform distribution for locations. We predict the location having the highest probability as the tweet's location. To remove any zero probability, we apply Laplace smoothing.

***tfidf* Value.** The naive Bayes model uses the probability of a word in each location when calculating the ranking probability of locations. If we want to reflect how important a word is in *all* locations, we can incorporate such global word weights by using the $tfidf$ values to rank the locations for a given tweet. Concretely, for a given tweet T, let L_T denote the set of candidate locations of T. We calculate the $tfidf$ value for each location l in L_T as follows:

$$tfidf_{T,l} = \frac{\sum_{w \in T} tfidf_{w,l}}{\sum_{l \in L_T} \sum_{w \in T} tfidf_{w,l}}$$

We use the location having the largest normalized $tfidf$ ranking score as the predicted location of tweet T.

4.3 Validating Step

Even though we can filter out some "I don't know" tweets using the local keyword filter and the PoS tag filter, sometimes the top-ranked candidate location for a tweet will fail to predict the actual location where the tweet was written. This is especially true for those tweets whose actual locations where the tweets were written are quite different from the referred location produced by our ranking algorithms. For example, we may think that the referred location in a real tweet "`Let's Go Yankees!!!`" is "Yankees Stadium" and some of our ranking techniques also find "Yankees Stadium" as the predicted location of the tweet. However, it is not unusual that many New York Yankees fans in the world post such tweets anywhere during the game or before the game. Another interesting real tweet is "`I hope you all have a GREAT weekend but also take time to remember those we've lost; those who are still fighting for our freedom!!`". Under an assumption that we know this tweet is from New York City, some of our ranking techniques find "World Trade Center" as the predicted location of the tweet. We can easily see that the tweet is closely related to "World Trade Center" semantically, however such tweets can be posted from anywhere. The main challenge for predicting the location for this type of tweets is to provide the prediction validation capability for the system to determine if the referred location $l_{ref}(T)$ for a tweet T, obtained using the probabilistic language models and one of the three ranking algorithms, will match the actual location $l_{cur}(T)$ where the tweet T was written. If we detect that $l_{ref}(T)$ does not match $l_{cur}(T)$, then we classify the tweet as an "unpredictable" tweet and exclude the tweet from our location prediction.

Our approach to finding such "unpredictable" tweets is to build a classification model using standard machine learning techniques. To learn the classification model, we need to prepare a training set carefully. One approach to preparing the training set is to use those tweets having a geo-tag (i.e., latitude and longitude), because such tweets already have their explicit current location, thus we can use the language models and one of the ranking algorithms to extract their referred location to build the training set. Given a tweet T having its geo-tag, after choosing the location (denoted as $l_{top}(T)$) having the highest probability based on the naive Bayes probability, we additionally compare the probability of $l_{top}(T)$ with that of the other locations using a probability ratio test. We use this test to build a good training set consisting of only tweets in which there is high confidence in their referred location. We choose only those tweets that pass the probability ratio test, formally defined as follows:

$$\frac{p(l_{ref}(T) \mid T)}{1 - p(l_{ref}(T) \mid T)} > \delta$$

where δ is the criterion of our test. If we increase δ, a smaller number of tweets will be selected for the training set.

Based on the generated training set, we learn classification models by running the decision tree classifier and SVM (Support Vector Machine) with the polynomial kernel functions and Gaussian radial basis functions using 10-fold cross-validation. Then we choose a classification model having the highest cross-validation precision for the training set and use this classification model for detecting the "unpredictable" tweets. To find parameters having the highest cross-validation precision, we use the grid search. We introduce some notable results returned by our classification model. For a real tweet "`The line at this Chipotle in Brooklyn Heights is really long`", our model detects that its referred location, produced by the language models and the ranking algorithm, indeed matches the actual location where this tweet was written, as indicated by the geo-tag of the tweet. Therefore, our model correctly classifies this tweet and thus validates the correctness of our predicted location of the tweet. Note that the accuracy of the prediction depends on our language models whereas the accuracy of the prediction validation depends on the training set.

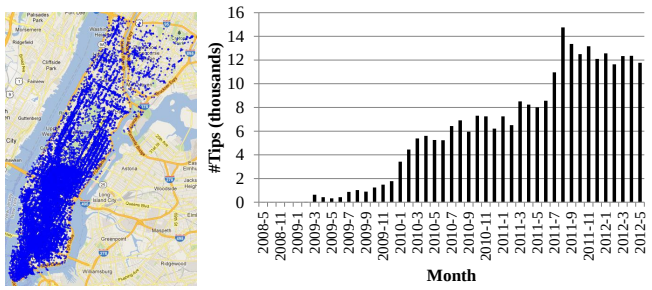
5. EXPERIMENTS

In this section, we evaluate the proposed location prediction framework for tweets through an extensive set of experiments conducted using tweets collected over a three-month period. We report the experimental results on how we build the language models using the datasets, how we implement the prediction validation classifier to distinguish the predictable tweets from those non-predictable ones, and the effectiveness of the two filters to find "I don't know" tweets. In addition, we evaluate the effectiveness of our location prediction approach by studying the effects of different parameters on the precision of location prediction, such as the effects of different ranking methods, the effects of unigram v.s. bigram language models, the effects of different δ values for building prediction validation classifier, and the effects of different $tfidf$ threshold values.

5.1 Datasets

We gathered a set of tweets spanning from April 2012 to June 2012 using Twitter Decahose [6] which is a feed of 10% of all tweets. Each day (24 hours) has about 37 million tweets and only 0.58% tweets are geo-tagged (i.e. include fine-grained location information). To focus on predicting the fine-grained location, we assume that we know the city-level (or similar) location of tweets because previous work [11, 19] has addressed this. Since some tweets explicitly include their city-level location even though they don't have their geo-tag, we can also utilize such information. In this paper, we select tweets from Manhattan, New York, USA because Manhattan (officially a borough of New York City), which covers 59 square kilometers (23 square miles), is one of the biggest and most densely populated cities in the world. Based on their geo-tag (latitude and longitude), 127,057 tweets (spanning three months) from Manhattan are selected. Among them, we exclude 39,157 tweets from Foursquare and 15,299 tweets from Instagram to remove any possible bias from them because they already include the location name in their textual data and so it would be straightforward to predict their location. Therefore, we use 72,601 tweets to evaluate our prediction framework.

We extracted Foursquare locations, called venues, and their tips using Foursquare API. First, to gather a set of Foursquare locations, we called the Foursquare venues API for each cell after splitting the area of Manhattan into very small cells (each covers 50 m $\times$ 50 m). Unfortunately, there were some missing locations using only this grid search. Therefore, to find additional locations, we analyzed the URLs included in check-in tweets from Foursquare and then extracted location IDs from them. Each Foursquare location has basic information such as name, address, latitude, longi-



(a) Locations in Manhattan

(b) #Tips in Manhattan by month

Figure 2: Foursquare locations and tips

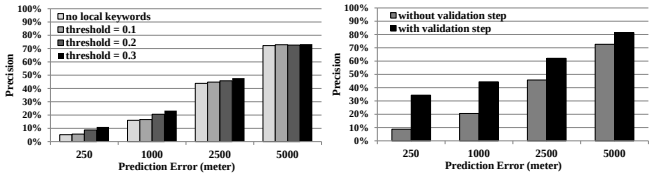
tude, city, country and category. Finally, for each gathered location, we extracted all its tips using Foursquare API. Using this approach, we gathered 25,171 venues in Manhattan and their 268,470 tips which span from May 2008 to June 2012. Also, there are some locations in which their area is too wide to represent their location using only one point, such as Central Park, Times Square and Yankee Stadium. Since Foursquare doesn't provide boundary information of its locations, we extracted boundary information of 22 wide locations in Manhattan using Google Maps. Figure 2(a) shows the geographical distribution of Foursquare locations in Manhattan and Figure 2(b) shows the distribution of total tips over the past 4 years, which shows a tremendous increase in the number of Foursquare tips in the last year.

5.2 Building language models

To build our language models for the extracted locations, we first choose locations which have more than 50 tips and so 1,066 locations are selected. We also experimented using language models of locations having more than 30 tips and 100 tips. However, the location prediction accuracy using them was not better than using locations having more than 50 tips. We believe that 30 or 40 tips are not enough to build a distinct language model for each location. On the other hand, for locations having more than 100 tips (e.g., 500 tips), we believe that the prediction accuracy will improve with more tips. However, there are only about 300 Foursquare locations in Manhattan having more than 100 tips and we think this number is too small to cover the area of Manhattan. Therefore, in this paper, we report results using language models of locations having more than 50 tips. For each location, to get a list of words from its tips, we first break each tip into tokens. Then we stem the tokens using Snowball stemmer [4] and remove any stop words in the tokens using stop words of Rainbow [21]. In addition to removing stop words, to consider only commonly used words for the location, we exclude words which appear in less than 5% tips among all tips of the location. Through this filtering, we can remove those words that are less common or contain typos, thus reduce the size of our word vocabulary (i.e., a set of all words used in our language models). Finally, 3,073 words are included in our word vocabulary.

5.3 Finding "I don't know" tweets

To find local keywords, we empirically choose three different $tfidf$ threshold values: 0.1, 0.2 and 0.3. For example, let us assume that a word appear in 10% of all locations (i.e. inverse document frequency, $idf = 1$). We can intuitively think that the word is too general to be included in the local keywords. By using 0.1 as the threshold, there should be any location in which the term frequency (tf) of the word is larger than 0.1 to be selected as a local keyword. Since it is rare for a word to occupy 10% of all tips, the word will be filtered out by the threshold. Table 1 shows the number of selected local keywords, among 3,073 words in our word vocabulary, for different $tfidf$ threshold values. To find tweets which are talking about the future or past, we utilize PoS tags generated by GPoSTTL [3].



(a) Without the validating step

(b) Effects of the validating step (threshold: 0.2)

Figure 3: Effects of the validating step

$tfidf$ threshold	# local keywords
0.1	1,782
0.2	556
0.3	200

Table 1: Local keywords

5.4 Prediction without the validating step

First we evaluate the prediction accuracy of our framework without applying the validating step for the predicted locations. To measure the prediction accuracy, given a tweet, we compare the geo-tag, which was removed during the prediction steps, of the tweet with the latitude and longitude (or boundary) of the predicted location. If the predicted location has its boundary information and the geo-tag of the tweet is within the boundary, the prediction error is 0. Otherwise, we calculate the Euclidean distance between the geo-tag of the tweet and the latitude and longitude of the location and then use the distance as the prediction error. We also note that acceptable prediction errors depend on the application in question. For example, automated geospatial review applications may require the location of the individual to be identified accurately (within 100m). On the other hand, applications such as event localization can tolerate a few hundreds of meters of error.

$tfidf$ threshold	# geo-tagged tweets	percentage
No local keywords	31,264	43.06%
0.1	28,057	38.65%
0.2	15,096	20.79%
0.3	7,168	9.87%

Table 2: Geo-tagged tweets without the validating step

Table 2 shows that our framework without the validating step can geo-tag a much more number of tweets, compared to 0.58% in the original Twitter data. However, as shown in Figure 3(a) where we use the naive Bayes model as the

ranking technique (we will compare different ranking techniques in the next section), the prediction precision is not satisfactory because only 10% of predicted tweets are located within 250m from their actual location even though we apply very selective local keywords (i.e., threshold = 0.3). Here, the precision means the percentage of predicted tweets whose prediction error is less than a specified distance (250m, 1,000m, 2,500m and 5,000m in Figure 3(a)). Although this result is meaningful compared to existing coarse-grained prediction frameworks, one of our goals is to improve the accuracy of our predicted locations. The results in subsequent sections show that we can considerably improve the prediction accuracy using our validating step.

5.5 Building models for the validating step

To validate the correctness of the predicted locations in terms of their likelihood to match the actual location where the tweets were written, we need to learn our classification models using the training datasets. In this set of experiments, we empirically use three different δ values: 0.5, 1.0 and 2.0 to generate three training sets. In other words, given a tweet, if there is a location whose naive Bayes probability is larger than 33%, 50% and 66%, the tweet will be included in the training set with the δ value of 0.5, 1.0 and 2.0 respectively. For each tweet, to label whether its referred location is equal to its current location, we compare the latitude and longitude of the referred location, extracted from Foursquare, with the geo-tag (i.e. current location) of the tweet. If the distance between the two locations is less than 100 meters or the geo-tag of the tweet is within the boundary of its referred location, we label that the tweet's two locations are the same. Table 3 shows the number of selected tweets, the number tweets whose two locations are different and the number of tweets whose two locations are the same, for different δ values among 72,601 tweets.

δ value	# tweets	# $l_{ref} \neq l_{cur}$	# $l_{ref} = l_{cur}$
0.5	2,642	1,936	706
1.0	1,598	1,008	590
2.0	1,028	579	449

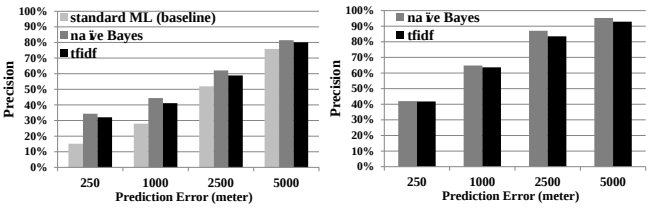
Table 3: Training sets

5.6 Prediction with the validating step

In this section, we first show the effectiveness of our classification-based prediction validation step for improving the prediction accuracy. Then we compare the location prediction accuracy by different ranking techniques and different parameter values. In this section, we use the $tfidf$ threshold of 0.2 and the δ value of 0.5, unless specifically noted, because we think this setting strikes a balance between the number of geo-tagged tweets and the prediction accuracy. We will show the effects of different parameter values in this section.

Effects of the validating step. Figure 3(b) shows that we can significantly improve the prediction precision using our validating step, compared to that without the validating step. Based on the generated classification model, by filtering out those tweets in which their predicted location does not match their actual location, we can locate about 34% of predicted tweets within 250m from their actual location.

Effects of different ranking techniques. Figure 4 shows the prediction precision of three different ranking



(a) Using only one prediction (b) Using the best in the Top-5

Figure 4: Effects of different ranking techniques

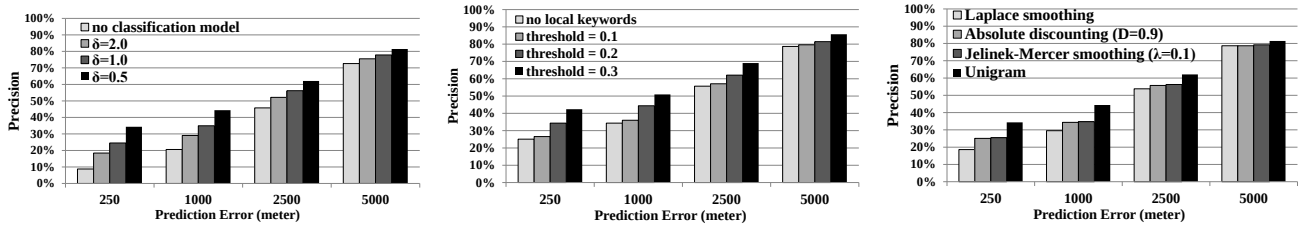
techniques on 2003 tweets predicted by our framework. We will show how 2003 tweets are predicted in the next experiment. Figure 4(a) shows that using the naive Bayes model as the ranking technique has better prediction precision than using standard machine learning techniques (our baseline approach) or $tfidf$ values. Specifically, using the naive Bayes model, about 34.35% and 44.38% of predicted tweets are located within 250m and 1,000m respectively from their location. This result shows that the naive Bayes model is working well in our language models to rank locations for given tweets even though the model does not consider global word weights. We think this is because our language models include only location-specific words (i.e. most of general words are filtered out by our local keywords and stop words). This may also be a reason that incorporating global word weights of such location-specific words, like $tfidf$ ranking, does not help much in terms of improving the prediction precision. In comparison, ranking with the standard machine learning (ML) techniques has relatively worse prediction precision because the prediction model is built using a very limited number of Foursquare tips. Since it is almost infeasible to use all (or most of) tips to run standard ML techniques due to the time complexity and the resource (CPU and memory) constraints, it would be hard to get good prediction results using this technique.

Figure 4(b) shows the prediction precision using the best prediction (i.e., the closest location from the geo-tag of tweets) in the top-5 predictions. This result represents the capacity of our prediction framework to find a set of good candidate locations even though the first predicted location is mistaken. The result shows that the naive Bayes model also has the best prediction precision by locating 41.99% of predicted tweets within 250m from their location. The prediction model generated using standard ML techniques has no top-5 result because it returns only one location having the highest confidence. Since the naive Bayes model has the best prediction precision in all other experiments using different parameter values, we report results using only the naive Bayes model in subsequent sections.

δ value	# geo-tagged tweets	percentage
0.5	2,003	2.76%
1.0	2,764	3.81%
2.0	3,982	5.48%

Table 4: Effects of different δ values

Effects of different δ values. We compare the number of tweets, among 15,096 tweets (See Table 2), classified as $l_{ref} = l_{cur}$ by different classification models built using different δ values in Table 4. The percentage in the table shows the ratio among 72,601 target tweets. Since the clas-



(a) Effects of different δ values (b) Effects of different $tfidf$ threshold values (c) Effects of bigram LMs

Figure 5: Effects of different parameter values

sification model using 0.5 as the δ value is built using the training set which includes more $l_{ref} \neq l_{cur}$ tweets compared to the other training sets as shown in Table 3, it has more capability to find such tweets and so choose fewer predictable tweets. The prediction precision result below shows that the classification model built using the δ value of 0.5 ensures higher precision by effectively filtering out unpredictable tweets. Figure 5(a) shows the prediction precision of our framework without any classification model and with three different classification models using different δ values. The prediction precision increases as the δ value decreases because, as we mentioned, the capability to filter out $l_{ref} \neq l_{cur}$ tweets increase due to the higher percentage of $l_{ref} \neq l_{cur}$ tweets in the training set. However, there would be a point in which selecting more tweets for learning the classification model by decreasing the δ value does not improve the prediction precision any more (or even worsens the prediction precision). This is because more noisy tweets which have low confidence in their referred location would be included in the training set by decreasing the δ value.

Effects of different $tfidf$ threshold values. Figure 5(b) shows the prediction precision of our framework without any local keywords and with three different $tfidf$ threshold values. Since the number of local keywords decreases as we increase the $tfidf$ threshold values as shown in Table 1, more tweets are filtered out as "I don't know" tweets because tweets should have at least one local keyword not to be excluded. Also, the precision continuously increases because selected tweets by high $tfidf$ threshold for the prediction have unique location-specific keywords. However, there is a trade-off between the prediction precision and the percentage of selected tweets. In other words, if we increase the $tfidf$ threshold to improve the prediction precision, a smaller number of tweets are selected for the prediction.

Unigram vs Bigram. In this section we compares unigram and bigram LMs under the same conditions. Figure 5(c) shows the prediction precision of bigram LMs with three different smoothing techniques and unigram LMs using the naive Bayes model. The effective smoothing parameters are selected from a coarse search of the parameter space. The result shows that unigram LMs are more effective than bigram LMs, which is consistent with the reported results [25]. This is because tweets and Foursquare tips are very short messages and it is rarely possible to include a bigram (or trigram or more), which can be used to effectively differentiate one location from another. Even though the location names include two or more words, the examination of prediction results verifies that unigram LMs are sufficient to detect such names. Also the effective parameters of absolute discounting and Jelinek-Mercer smoothing shows that the smoothed bigram LMs work better when they assign more weights on unigram LMs.

Approach	Percentage
original Twitter data	0.72%
original Twitter data (excluding Foursquare & Instagram)	0.58%
our framework (without validation step)	20.79%
our framework (with validation step)	2.76%

Table 5: Percentage of geo-tagged tweets

5.7 Percentage of geo-tagged tweets

Finally we summarize how many tweets are geo-tagged by our prediction framework in Table 5. This result indicates how well our framework tackles the location sparseness problem of Twitter. In the original Twitter data, only 0.72% tweets have their geo-tag. For fair comparison with our framework in which we exclude tweets from Foursquare and Instagram because it is too trivial to predict their location, the percentage of geo-tagged tweets in the original Twitter data goes down to 0.58% if we don't count the tweets from Foursquare and Instagram. We report in this section the results of our framework using the δ and $tfidf$ threshold value of 0.5 and 0.2 respectively and the naive Bayes model as the ranking technique because we think this setting strikes a balance between the number of geo-tagged tweets and the prediction accuracy. Our framework equipped with all proposed techniques including the validating step can geo-tag 2.76% of all tweets, increasing about 4.8 times compared with the percentage of geo-tagged tweets in the original Twitter data, while placing 34% of predicted tweets within 250m from their actual location. If we don't use our classification-based prediction validating method, we can geo-tag 20.79% of all tweets with lower prediction accuracy as shown in Table 2.

6. CONCLUSION AND FUTURE WORK

We have addressed the location sparseness problem of tweets by developing a framework for increasing the number of geo-tagged tweets by predicting the fine-grained location of each tweet using only textual content of the tweet. Our framework is vital for many applications which require more geo-tagged tweets such as location-based advertisements, entertainments and tourism. Our prediction framework has two unique features. First of all, we build the probabilistic language models for locations using unstructured short messages that are tightly coupled with their locations in Foursquare, instead of using noisy tweets. Second, based

on the probabilistic models, we propose a 3-step technique (Filtering-Ranking-Validating) for tweet location prediction. In the filtering step, we develop a set of filters that can remove as many location-neutral tweets as possible to minimize the noise level and improve the accuracy of our location prediction models. In the ranking step, we utilize ranking techniques to select the best candidate location as the predicted location for a tweet. In the validating step, we develop a classification-based prediction validation method to ensure the correctness of predicted locations. Our experimental results show that our framework can increase the percentage of geo-tagged tweets about 4.8 times compared to the original Twitter data while locating 34% of predicted tweets within 250 meters from their location. To the best of our knowledge, this is the first work which incorporates external data source such as Foursquare, in addition to Twitter data, for location prediction of each tweet. Furthermore, unlike most existing frameworks which focus on coarse-grained prediction such as 10km and 100km, our framework locates a considerable amount of predicted tweets within one-quarter kilometer from their location.

It should be noted that, for privacy advocates, our results can be interpreted as new threats to location privacy for their short messages such as tweets. In other words, our techniques can be used not only to provide the valuable geo-tag information of tweets for location-based services but also to give warning of potential risks to their location to the privacy advocates. For example, when a Twitter user, who is concerned about his/her privacy, posts a tweet, our framework can detect that the location of the tweet can be predicted with high confidence and give him/her a warning of potential threats to location privacy. Our framework can also provide real-time warnings, while the user is writing a tweet, by checking whether the newly entered word is included in the local keywords.

Even though the focus of this paper is exploring location-specific information explicitly or implicitly included in the textual content of tweets, our framework can be extended by incorporating more information sources to further increase the number of geo-tagged tweets and improve the location prediction accuracy. One simple extension could be to build time-based models (per day, week, month and year) for each location and then utilize the models with the timestamp of a given tweet to predict its location. For example, if our time-based models for a museum indicate that there is almost no activity after 6pm on weekdays, our prediction framework would give very low ranking to the museum for a tweet which was posted at 9pm on a Wednesday. Another possible extension could be to consider a set of tweets, including Foursquare check-in tweets, posted by a single user as time series data. This information could be used to fine-tune the prediction of our framework. For example, if a user posted a Foursquare check-in tweet, we can reduce the search space for predicting the location of those tweets, posted by the same user and whose timestamp is close to that of the Foursquare tweet. Furthermore, if a user posted two Foursquare check-in tweets at two different locations within a short period of time, we could predict the location of those tweets posted between the two timestamps of the Foursquare tweets by analyzing the possible trajectory paths between the two locations using some interpolation techniques, like the route matching algorithm [28]. Other interesting extensions to our current framework includes inference over future and past activities included in the tweets, utilizing social relationships between Twitter users, spatial and temporal relationship as well as semantic relationship among different tweets.

7. REFERENCES

[1] About foursquare. https://foursquare.com/about/.

[2] Geo Developer Guidelines. https://dev.twitter.com/terms/geo-developer-guidelines.

[3] GPoSTTL. http://gposttl.sourceforge.net/.

[4] Snowball. http://snowball.tartarus.org/.

[5] Twitter: A New Age for Customer Service - Forbes. http://onforb.es/VqqTxa.

[6] Twitter Decahose. http://gnip.com/twitter/decahose.

[7] E. Amitay, N. Har'El, R. Sivan, and A. Soffer. Web-a-where: geotagging web content. In *Proceedings of the 27th annual international ACM SIGIR conference on Research and development in information retrieval*, SIGIR '04, pages 273–280, New York, NY, USA, 2004. ACM.

[8] L. Barbosa and J. Feng. Robust sentiment detection on twitter from biased and noisy data. In *Proceedings of the 23rd International Conference on Computational Linguistics: Posters*, COLING '10, pages 36–44, Stroudsburg, PA, USA, 2010. Association for Computational Linguistics.

[9] F. Benevenuto, G. Magno, T. Rodrigues, and V. Almeida. Detecting spammers on twitter. In *Proceedings of the 7th Annual Collaboration, Electronic messaging, Anti-Abuse and Spam Conference (CEAS)*, 2010.

[10] S. F. Chen and J. Goodman. An empirical study of smoothing techniques for language modeling. In *Proceedings of the 34th annual meeting on Association for Computational Linguistics*, ACL '96, pages 310–318, Stroudsburg, PA, USA, 1996. Association for Computational Linguistics.

[11] Z. Cheng, J. Caverlee, and K. Lee. You are where you tweet: a content-based approach to geo-locating twitter users. In *Proceedings of the 19th ACM international conference on Information and knowledge management*, CIKM '10, pages 759–768, New York, NY, USA, 2010. ACM.

[12] Z. Cheng, J. Caverlee, K. Lee, and D. Sui. Exploring millions of footprints in location sharing services. In *International AAAI Conference on Weblogs and Social Media*, ICWSM '11, 2011.

[13] B. Hecht, L. Hong, B. Suh, and E. H. Chi. Tweets from justin bieber's heart: the dynamics of the location field in user profiles. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '11, pages 237–246, New York, NY, USA, 2011. ACM.

[14] Y. Ikawa, M. Enoki, and M. Tatsubori. Location inference using microblog messages. In *Proceedings of the 21st international conference companion on World Wide Web*, WWW '12 Companion, pages 687–690, New York, NY, USA, 2012. ACM.

[15] R. Jones, R. Kumar, B. Pang, and A. Tomkins. "i know what you did last summer": query logs and user privacy. In *Proceedings of the sixteenth ACM conference on Conference on information and knowledge management*, CIKM '07, pages 909–914, New York, NY, USA, 2007. ACM.

[16] A. Kavanaugh, S. Yang, S. D. Sheetz, and E. A. Fox. Microblogging in crisis situations: Mass protests in iran, tunisia, egypt. In *Workshop in conjuction with the ACM Conference on Human Factors in Computing Systems (CHI'11)*, 2011.

[17] W. Li, P. Serdyukov, A. P. de Vries, C. Eickhoff, and M. Larson. The where in the tweet. In *Proceedings of the 20th ACM international conference on Information and knowledge management*, CIKM '11, pages 2473–2476, New York, NY, USA, 2011. ACM.

[18] J. Lin, R. Snow, and W. Morgan. Smoothing techniques for adaptive online language models: topic tracking in tweet streams. In *Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining*, KDD '11, pages 422–429, New York, NY, USA, 2011. ACM.

[19] J. Mahmud, J. Nichols, and C. Drews. Where is this tweet from? inferring home locations of twitter users. In *International AAAI Conference on Weblogs and Social Media*, ICWSM '12, 2012.

[20] C. D. Manning, P. Raghavan, and H. Schtze. *Introduction to Information Retrieval*. Cambridge University Press, New York, NY, USA, 2008.

[21] A. K. McCallum. Bow: A toolkit for statistical language modeling, text retrieval, classification and clustering. http://www.cs.cmu.edu/ mccallum/bow, 1996.

[22] A. Noulas, S. Scellato, C. Mascolo, and M. Pontil. An empirical study of geographic user activity patterns in foursquare. In *International AAAI Conference on Weblogs and Social Media*, ICWSM '11, 2011.

[23] M. Pennacchiotti and A.-M. Popescu. Democrats, republicans and starbucks afficionados: user classification in twitter. In *Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining*, KDD '11, pages 430–438, New York, NY, USA, 2011. ACM.

[24] T. Sakaki, M. Okazaki, and Y. Matsuo. Earthquake shakes twitter users: real-time event detection by social sensors. In *Proceedings of the 19th international conference on World wide web*, WWW '10, pages 851–860, New York, NY, USA, 2010. ACM.

[25] F. Sebastiani. Machine learning in automated text categorization. *ACM Comput. Surv.*, 34(1):1–47, Mar. 2002.

[26] P. Serdyukov, V. Murdock, and R. van Zwol. Placing flickr photos on a map. In *Proceedings of the 32nd international ACM SIGIR conference on Research and development in information retrieval*, SIGIR '09, pages 484–491, New York, NY, USA, 2009. ACM.

[27] K. Starbird and L. Palen. (how) will the revolution be retweeted?: information diffusion and the 2011 egyptian uprising. In *Proceedings of the ACM 2012 conference on Computer Supported Cooperative Work*, CSCW '12, pages 7–16, New York, NY, USA, 2012. ACM.

[28] A. Thiagarajan, J. Biagioni, T. Gerlich, and J. Eriksson. Cooperative transit tracking using smart-phones. In *Proceedings of the 8th ACM Conference on Embedded Networked Sensor Systems*, SenSys '10, pages 85–98, New York, NY, USA, 2010. ACM.