When a Tweet Finds its Place: Fine-Grained Tweet Geolocalisation

Pavlos Paraskevopoulos[1] and Giovanni Pellegrini[2] and Themis Palpanas[3]

[1] University of Trento
Telecom Italia - SKIL
p.paraskevopoulos@unitn.it
[2] University of Trento
Telecom Italia - SKIL
giovanni.pellegrini@studenti.unitn.it
[3] Paris Descartes University
themis@mi.parisdescartes.fr

Abstract. The recent rise in the use of social networks has resulted in an abundance of information on different aspects of everyday social activities that is available online. In the process of analysis of the information originating from social networks, and especially Twitter, an important aspect is that of the geographic coordinates, i.e., geolocalisation, of the relevant information. This information is used by a variety of applications for the better understanding of an urban area, the tracking of the way a virus spreads, the identification of people that need help in case of a disaster (e.g., an earthquake), or just for the better understanding of the dynamics of a major event (e.g., a concert). However, only a tiny percentage of the twitter posts are geotagged, which restricts the applicability of location-based applications. In this work, we extend our framework for geolocating tweets that are not geotagged, and describe a general solution for estimating the city and neighborhood in the city, from which a post was generated. In addition, we study the specific problem of geolocalising tweets deriving from targeted locations of interest (i.e., cities and neighborhoods in these cities), and present the visualizations of the prototype dashboard application we have developed, which can help end-users and large-scale event organizers to better plan and manage their activities. The experimental evaluation with real data demonstrates the efficiency and effectiveness of our approach.

Keywords: geotag, geolocation, Twitter, social networks

1 Introduction

[**Motivation:**] Events that happen around us affect our lives to different degrees. The effects of an event on a community vary depending on the type of the event and its dynamics. For example, traffic jams affect the way we move, football matches and concerts may affect the normal pace of life in the area of the venue for a short period of time, while earthquakes and diseases are unpredicted events, which could cause significant problems that have to be addressed fast. Many entities, public and private, are interested in analyzing the effects of such events, in order to better understand and react to them, and lead to a better quality of life. For example, the identification of lack of clean water at a place would lead the water providers to take special care for resolving the problem. Even though this would be a manual, labour-intensive, and time-consuming process in the past (e.g., consider the 1854 cholera outbreak in London [17]), this is no longer the case.

People tend to share their experiences, especially those affecting their lives (or feelings). Social networks, such as Twitter [3], Facebook [1] and Google+ [2], give users the opportunity to express themselves and report details about their everyday social activities. The combination of this behavior with the widespread use of mobile smart-phones and tablets has allowed users to report their activities in real time, adding reports from several different locations (not just from their homes, or workplaces). Consequently, we now have access to datasets containing detailed information of social activities. To that effect, several studies [29], including applications [20, 32, 14, 7, 11, 6, 30, 35] and techniques [33, 28, 23, 31] have been developed that analyze datasets created through the use of social networks, tracking crowd movements and identifying needs, in order to provide benefits to end users, businesses, civil authorities and scientists alike.

It is interesting to note that several of these applications depend on the knowledge of the user location at the time of the posting. This knowledge is necessary for applications that target to characterize an urban landscape, or to optimize urban planning [14], to monitor and track mobility and traffic [7], and to identify and report natural disasters, such as earthquakes [11]. For example, in the case of earthquakes knowing the exact location of a tweet can provide actionable insights to emergency-response workers (extent of damages, or number of victims at specific locations, etc.) [8]. Such applications, which represent an increasingly wide range of domains, are restricted to the use of geotagged data[4], that is, posts in social networks containing the geographic coordinates of the user at the time of posting.

Evidently, the availability of geotagged data, determines not only the possibility to use such applications, but also their quality-performance characteristics: the more geotagged data posts are available, the better the quality of the results will be (more precisely: the higher the probability for being able to produce better quality results). Nevertheless, the availability of geotagged data is rather limited. In Twitter, which is the focus of our study, the number of geotagged tweets is a mere 1.5-3% of the total number of tweets [19, 21, 15]. As a result, the amount of useful data for these applications to analyze is small, which in turn limits the utility of the applications. Even if we considered this subset of geotagged tweets as representative, "there is a tendency for geotaggers to be

[4] For the rest of this paper, we will use the terms *geotagged* and *geolocalised* interchangeably.

slightly older than non-geotaggers" [27], which may lead to non-representative, or skewed results.

[**Proposed Approach and Contributions:**] In this study, we address this problem by extending our framework [24] for geolocalising tweets that are non-geotagged. Even though previous works have recognized the importance and have studied this problem [9, 18] (for a comprehensive discussion of this problem refer to [15]), their goal was to produce a coarse-grained estimate (i.e., postal zipcodes, cities, or geographical areas larger than cities) of the location of a set of non-geotagged tweets (e.g., those originating from a single user).

In contrast, in our previous work [24], we examined this problem at a much finer granularity, thus enabling a new range of applications that require detailed geolocalised data. More specifically, our solution provides location estimates for *individual* tweets, at the level of a *city neighborhood* given the city (or the city, given the country). That is, we focused on the identification of the location, where the location belonged to a set of candidate locations. This solution exploits the similarities in the content between an individual tweet and a set of geotagged tweets, as well as their time-evolution characteristics.

In this work, we extend our previous solution, and describe a general technique for estimating the location from which a post was generated using a two-stage process: we first determine the city, and then the neighborhood in the city, by building content-based models and analyzing the volume of posts over time, independently for each one of these two levels. Using this set up, we are able to effectively predict the location of a post form the Twitter stream, when the only input we have is the actual content of the post and its timestamp.

In addition, we study the specific problem of geolocalising tweets deriving from targeted locations of interest, that is, neighborhoods of a particular cultural, social, or touristic importance (e.g., the Vatican in Rome). Our experiments show that we can reuse our technique for this case, as well, by adjusting its operation to this context, where a small number of popular keywords mentioned in the posts characterize the location.

Finally, we present the visualizations of the prototype dashboard application we have developed, which can help end-users and large-scale event organizers to better plan and manage their activities. These interactive visualizations include heatmaps for the volume of (geotagged and geolocalised) tweets, where the user can zoom at different levels of granularity, ranging from a country, down to a city neighborhood, for which the user can also explore the relevant keywords. Furthermore, we provide visualizations that illustrate in a comprehensive manner the changes in the volume of posts at different locations over time.

[**Paper Organization:**] The rest of the document is organized as follows. In Section 2 we present the related work. Section 3 formalizes the problem, and Section 4 describes our solution. We present our experimental evaluation in Section 5, and our prototype dashboard implementation in Section 6. Finally, we conclude in Section 7.

2 Related Work

Several works have studied the problem of geotagged tweet analysis. Balduini et al. [7] studied the movement of people by analyzing geotagged tweets. Some studies focus on the extraction of local events by analyzing the text in the tweets [12]. Abdelhaq et al. [4] use both geotagged and non-geotagged tweets for identifying keywords that best describe events. We note that in all the above studies, the tweets that are analyzed are already geotagged. In contrast, our focus is on non-geotagged tweets.

The problem of using tweets in order to identify the location of a user, or the place that an event took place has been studied in the past. The "who, where, what, when" attributes extracted from a user's profile can be used to create spatio-temporal profiles of users, and ultimately lead to identification of mobility patterns [34]. Cheng et al. [10] create location profiles based on idiomatic keywords and unique phrases mentioned in the tweets of users who have declared those locations as their origins.

The similarity between user profiles and location profiles has also been used in [9], where they compute a set of representative keywords for each location, which allows the algorithm, to compute the probability that a given user comes from that location. Furthermore, the authors of [15], have evaluated the methods using as test datasets either geotagged, or non-geotagged tweets, showing that "a model trained on geotagged data indeed generalizes to non-geotagged data".

Two recent approaches that target to geotag unique tweets are presented in [16] and [25]. The main difference to our approach is that these methods rely on users that post many tweets in a time interval t, or on data from the user's profile. In contrast, we target to geotag tweets even from users that have never posted before, or do not provide any profile data (such as their home location).

Two studies that target to geotag tweets are presented in [13] and [22]. These two methods create chains of words that represent a location. The latter study takes in addition into consideration the location a user has recorded as their home location. A study that predicts both a user's location and the place a tweet was generated from is presented in [18]. In this study, the authors construct language models by using Bayesian inversion, achieving good results for the country and state level identification tasks. Finally, [26] presented a method for identifying the geolocation of photos by using the textual annotations of these photos.

Even though some of these studies are closely related to our work (e.g., [9, 18], we observe that they operate at a very different time and space scale. The profiles they create involve the tweets generated over a long period of time (up to several months), and the location that has to be estimated is the location of origin of the user, rather than the location from where a particular tweet was posted. Moreover, the space granularity used in these studies ranges from postal zipcodes to areas larger than a city. On the contrary, in our work we predict the location of individual tweets, at the level of city neighborhoods, and our approach has been shown to achieve the best results when compared to the state of the art [24]. A recent survey presents methods relevant to location inference [5].

3 Problem Formulation

The problem we address in this work is the estimation of the geographic location of individual, non-geotagged posts in social networks.

Problem 1: Given a set of geotagged posts $P^{l_1}_{t_j}, ..., P^{l_i}_{t_j}$, $t_1 \leq t_j \leq t_2$, where l_i is the location the post was generated from and t_j is the time interval during which the post was generated at, and a set of individual non-geotagged posts ${Q^1}_{t_q}, ..., {Q^n}_{t_q}$, $t_1 \leq t_q \leq t_2$, we want to identify the location l from which the post ${Q^i}_{t_q}$ $(1 \leq i \leq n)$ was generated.

The timestamps t_1 and t_2 represent the start and end times, respectively, of the time interval we are interested in.

In the context of this work, we concentrate on two-level fine-grained location predictions: we wish to initially estimate the coarse-grained location of a post (which is usually as big as a city) and afterwards to estimate the location at a much finer-grained level such as a city neighborhood (which is usually much smaller than a postal zipcode). Furthermore, we focus on twitter posts, whose particular characteristics are the very small size (i.e., up to 140 characters long), and the heavy use of abbreviations and jargon language.

4 Proposed Approach

In this section, we describe our solution to the problem of fine-grained geolocalisation of non-geotagged tweets. Our method is based on the creation of vectors describing the Twitter activity, in terms of important keywords, for each geolocation we have data from, and for the period of time we are interested in.

In this paper we modify the set up of our method, extending this work so that we can use tweets from the entire stream of a social network (refer to Algorithm 1). This is an extension of our previous work (details can be found in [24]), where we apply the search in two stages: initially using a coarse granularity (such as a city), and subsequently, a fine granularity (such as a city neighborhood). We additionally introduce dynamic data-driven similarity thresholds that get automatically readjusted, which can lead to high precision (as we demonstrate in Section 5).

[**Extraction of Important Keywords and Creation of Location Vector:**] We initialize our method by defining the Coarse-Grained Locations (CGL) that we target to identify posts from (i.e., cities), and the time intervals we are interested in. Afterwards, we get the geotagged posts generated from the CGLs in our predefined time intervals, and we group them according to the CGL they were generated from. We then follow exactly the same steps as described in [24] for extracting the keyword-vector of each location. The only difference in our current algorithm is that we do not need to keep the concordance vectors at the end, since we only use the Tf-Idf vectors.

Having created the CGL vectors, we repeat the procedure for the Fine-Grained Locations (FGL), where each FGL is a subset of a CGL (e.g., a city

Algorithm 1 Tweet Geotagging Algorithm

INPUT: A training set of timestamped and geotagged tweets, a timestamped query-tweet (Q_t) that is not geotagged, a set of predefined coarse-grained locations (CGL) and a set of predefined fine-grained location (FGL) where $CGL \subset FGL$

OUTPUT: The most eligible candidate location.

```
 1: $kwVector_{Q_t} \leftarrow$ create vector of $Q_t$ keywords and their weights    $\triangleright$ process
    non-geotagged tweet $Q_t$
 2: for all $i \in \{CGL\}$ do   $\triangleright$ process training dataset, for all coarse-grained locations
 3:     for all $t \in \{\text{time intervals}\}$ do                $\triangleright$ and for all time intervals
 4:         $Doc_{i_t} \leftarrow$ all tweets in location $i$ at time interval $t$
 5:         $kwVector_{i_t} \leftarrow$ create vector of $Doc_{i_t}$ keywords and their weights
 6:     $similarity_i \leftarrow VectorSim(kwVector_{Q_t}, kwVector_{i_t})$
 7:     if $similarity_i > 0$ then
 8:         Add CGL to candidate coarse grained locations (CandCGL)
 9: for all $j \in CandCGL$ do
10:     for all $FGL_i \in j$ do
11:         for all $t \in \{\text{time intervals}\}$ do            $\triangleright$ and for all time intervals
12:             $Doc_{FGL_{i_t}} \leftarrow$ all tweets in location $FGL_i$ at time interval $t$
13:             $kwVector_{FGL_{i_t}} \leftarrow$ create vector of $Doc_{FGL_{i_t}}$ keywords and their weights
14:         $similarity_{FGL_i} \leftarrow VectorSim(kwVector_{Q_t}, kwVector_{FGL_{i_t}})*$
15:         $TwoLevelSimilarity_{FGL_i} \leftarrow similarity_j * similarity_{FGL_i}$
16: $location \leftarrow argmax_{i \in FGLs}\{ProbCalc(TwoLevelSimilarity)$ $\triangleright$ identify location of
    tweet $Q_t$
17: return $location$
```

neighborhood), extracting the keyword vectors that describe the activity for the level of the FGLs.

[**Similarity Calculation and Best Match Extraction:**] We then calculate the similarity between the keyword vector of Q and the keyword vector of each one of the CGLs, using the function $VectorSim$ (as described in [24]). Although up to this point the similarity extraction is the same as in our base-method, we note that it is possible to have posts that are not generated from the predefined locations. Yet, they could have a small similarity with our candidate locations, leading the algorithm to wrongly assign them to our candidate CGL. In order to avoid such cases, we use dynamic data-driven thresholds that allow us to filter out these posts (we describe this procedure in detail in Section 4.1).

Having extracted the most eligible CGLs, we proceed to the next stage and check all the FGL that are subsets of an eligible CGL (once again using the $VectorSim$ function). Although we have already filtered the CGLs with low similarity, it is possible to encounter low-similarity scores at the level of FGL, as well. In order to avoid this, we use an additional data-driven threshold, and we extract the FGLs that exceed the threshold of the fine-grained granularity. We then combine the FGL similarity with the CGL similarity, multiplying them and getting a unique value for each eligible FGL, which we normalize and convert

Algorithm 2 Probability Calculation

```
1: procedure ProbCalc(similarities between $Q_t$ and candidate geolocations ($Geolocs$))
2:     for all $i \in Geolocs$ do                ▷ Get the probability distribution
3:         $Prob_{i_t,Q_t} \leftarrow \frac{Sim_{i_t,Q_t}}{\sum Sim_{Q_t}}$
4:     SortDescending $Prob_{i_t,Q_t}$
5: return $Geolocs$ and their $Prob_{i_t,Q_t}$
```

into a probability. At the end, the algorithm returns the geolocation with the highest probability (refer to Algorithm 2).

[**Similarity Based on Correlation of Activity Time Series:**] As established in our previous study [24], the method that achieves the best results is the TG-TI-C (Tweet Geotag with the use of Tf-Idf and Correlation). This method, apart from the content similarity that we presented above, also uses the correlation factor for the calculation of the similarities.

The correlation factor is a parameter that allows us to exploit the time-evolution behavior of a location: we initially extract the correlation between "global" and "local" locations, and afterwards we multiply it to the similarity extracted between the $keywordVector_{local}$ and the $keywordVector_{Q}$. in our current study, we employ this feature, but restrict it only to FGLs, since (as we explained earlier) we are now dealing with posts that may not correspond to any of the candidate locations.

4.1 Dynamic Threshold Extraction

As we have already mentioned, the set up of this method allows us to identify non-geotagged tweets that are coming from the full stream of a social network. As a result, posts irrelevant to our candidate locations could still share stopwords, leading to a (small) similarity to some location. In order to filter out these cases, we use thresholds on the similarity, both for the CGLs and the FGLs.

The distribution of the keywords among the candidate locations is different depending on the time intervals we check. Therefore, the significant keywords are going to have different weights for each time interval. For example, during the night we have a few posts, leading to the creation of small dictionaries, where matching one of the stopwords in these dictionaries would lead to high similarity between the $Q - tweet$ and the candidate location. In this case, the threshold should be set high. This is not true when we consider the dictionaries created during the day.

In order to automatically set a dynamic threshold, we use a small training dataset (in our case 1 day), keeping the similarities between each $Q - tweet$ and the location that it corresponds to. We initiate our threshold by setting it to 0. Then, we identify the tweets that are correctly matched to a location, and we record their similarity. At the end, we compute the mean of all the similarities, giving us the threshold extracted from the first day and for the specific time intervals. In order to set up the threshold for these time intervals for the following day, we use the mean of the thresholds used in all previous days. As a result, the threshold for a given day and time interval is computed as the mean of the threshold means of the previous days for the same time interval.

Following the procedure described above, we dynamically update the threshold: the thresholds are data driven, and the method is parameter-free.

5 Experiments

5.1 Experimental Setup

We performed the experiments on a PC with an Intel Core i7-2600 CPU @ 3.40GHz x 8 processor, running Ubuntu 16.04.2 LTS with 8GB RAM. We implemented our algorithms in Python 2.7, and used the Geoplotlib toolbox for the visualizations.

[**Datasets:**] We use a real dataset containing geotagged posts from Twitter, generated in Italy between June 1 and June 20, 2016. The coarse-grained locations that we focus on are the 7 Italian cities with the highest activity, namely Rome, Milan, Florence, Venice, Naples, Turin and Bologna. The total number of tweets is 218,572: 23,566 originated from Rome, 18,824 from Milan, 7,840 from Florence, 4,628 from Venice, 4,071 from Naples, 3,719 from Turin, 2,624 from Bologna and 153,300 from the rest of the country. As fine-grained locations, we consider city neighborhoods represented by 1km-side squares. For targeted locations, we have selected the Vatican (a 1.3km-side square) from Rome, and San Siro Stadium (a 0.8km-side square) from Milan. The time windows we use have a duration of 4 hours (which can effectively capture an important event, as well as the start and the aftermath of this event), while also keeping the detailed aggregated information for every 15min time interval.

Instead of using tumbling windows as done in [24], we use sliding windows that lead to more up-to-date topic models for our locations. We experimented sliding the window by 1 and by 2 time intervals, getting almost the same results; thus, we chose to slide our window by 2 time intervals per slide (30-minutes), which led to faster execution times. Finally, we update the keywordVectors incrementally at every slide, which means that our method can be used in an online fashion. In this experiment, we had 1920 15-min timeslots, resulting into 952 window slides.

[**Algorithms:**] We experimentally evaluate the two-level algorithm we described in Section 4. In order to initialize our thresholds, we run our method on the first day, but exclude these results from the evaluation. In all our experiments, we randomly divided the dataset in 80%training and 20% testing, repeated each experiment 5 times, and reported the mean values in the results. We compare the results of our two-level algorithm with the state-of-the-art QL and KL algorithms [18] (for a more detailed comparison, refer to [24]).

[**Evaluation Measures:**] We study the effectiveness of our approach using the precision and recall measures: $Precision = \frac{cgTweets}{gTweets}$ and $Recall = \frac{cgTweets}{aTweets}$, where $cgTweets$ is the number of the correctly geolocalised tweets, $gTweets$ is the number of tweets we geolocalised, and $aTweets$ is the number of all tweets that are originally deriving from our candidate locations. We also report the balanced F1 measure, $F1 = 2 * \frac{Precision*Recall}{Precision+Recall}$.

5.2 Performance with Varying Number of CGLs

In the first set of experiments, we study the performance of our method when we vary the number of CGLs (i.e., cities) between 1 and 7. As estimated location, we only consider the first answer given by our algorithm (i.e., @Top1). We note that the random algorithm had precision less than 0.024% and recall less than 0.12%, with the highest values occurring when using 1 city.

In Figures 1a-1b, we illustrate the precision and recall when we use 7 CGLs. The results for 1-6 CGLs are very similar, and omitted for brevity. The F1 for the cases of 1 and 7 CGLs are compared in Figures 1c and 1d, respectively. Using our approach, we achieve a precision of up to 89%, and a recall of up to 17%, while the best F1 was 26%. The best precision is achieved when using 60% of the keywords and a threshold of +20%, while the best recall and F1 for 70% of the keywords and "no threshold".

For the comparison to the state-of-the-art presented in Figure 2, we use the version of our method with threshold +20%. As depicted in the plots, our method achieves up to 80% precision and 23% recall, while KL only achieves up to 13% precision and 20% recall.

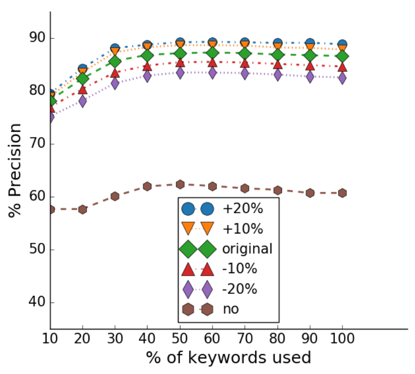
We note that as we increase the number of CGLs considered, we would expect to see a reduction in the precision and recall values, as a result of the increased search space. When looking at all the detailed results though, we do not observe this. On the contrary, precision slightly increases as we add CGLs, demonstrating the robustness of our approach.

5.3 Performance for Targeted Locations

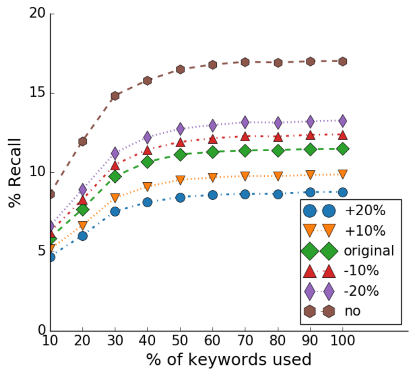
We now evaluate the performance of the proposed approach for targeted locations of interest. The results for the Vatican and San Siro locations are presented in Figures 3a-3b, and Figures 3c-3c, respectively.

The precision for Vatican reaches a maximum of 68% when using either 10%, or 20% of the keywords and "no threshold", while recall reaches 84% when using 100% of the keywords and "no threshold". Similarly, San Siro achieves a precision of 49%, and a recall of 54%, for 10%, or 20% of the keywords, and for 100% of the keywords, respectively, and "no threshold". These numbers correspond to a pretty high performance, especially when taking into account the very high recall values.

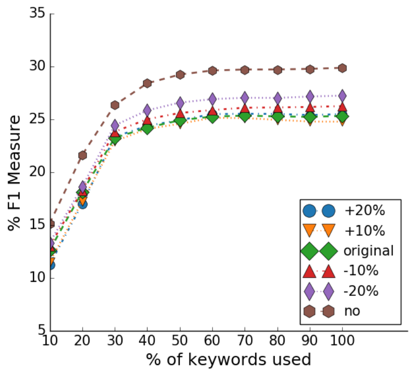
We note that in both locations, the precision and recall values are exactly the same when using 10% and 20% of the keywords, while the precision reduces suddenly after that. A close look at the dictionaries of the two locations revealed that the most important keywords are the names of the locations. The small



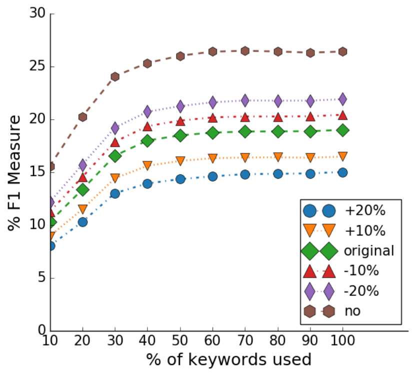
(a) Precision for 7 CGLs



(b) Recall for 7 CGLs



(c) F1 for 1 CGL (Milan)



(d) F1 for 7 CGLs

Fig. 1: (Top) Trade-off Between Precision and Recall for 7 CGLs (Rome, Milan, Venice, Florence, Naples, Bologna, Turin, @Top1). (Bottom) F1 for 1 and 7 CGLs (@Top1).

dictionary size employed (when using 10-20% of the keywords) is then occupied by these keywords. As the dictionary size increases, stopwords and noise are inserted, which have a negative impact on precision.

6 Dashboard

We now provide a brief description and sample screenshots from our prototype dashboard implementation, which uses our tweet geolocalisation solution in order to help users visualize and better understand geolocalized Twitter activity.

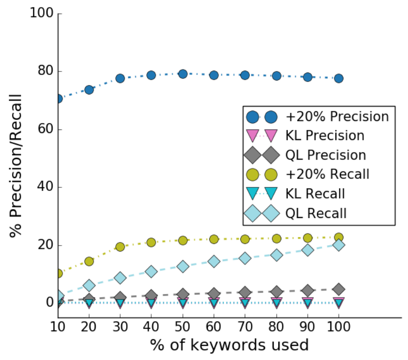
Figure 4a depicts the geotagged tweets that are generated from Italy in the form of a heat-map (the more red the color of a location, the more tweets originate from that location). Due to the bounding box used for extracting the tweets, we have tweets from places outside of Italy, as well. Nevertheless, this does not pose a problem, since our approach aims to geotag any tweet in the public stream, and is language independent. As we can observe from the heat-map, the Twitter activity is heavy in the big cities (e.g., Milan and Rome), and low in the country side.

The user can then use the map to zoom in, for example on the capital of Italy, Rome. In this case, we observe that the activity around important locations, such as the Vatican and the Colosseum, is much higher than the activity at the suburbs of the city (refer to Figure 4b).

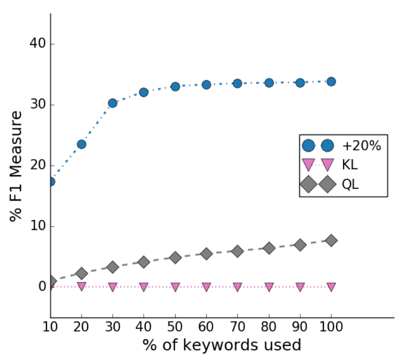
If the user continues to zoom in, they arrive at the detailed city view, illustrated in Figure 5a. In this view, we can see the square grid structure, superimposed with a heat-map that represents the distribution of the geolocalised tweets (the more tweets a location has, the more dense the color is). In addition, this view allows the user to examine the topic models computed for each one of the neighborhoods: when the mouse hovers over a square, a window pops up that lists the most important keywords for that location.

Another useful analysis tool is the study of how the volume of tweets changes over time in different neighborhoods of a given city. This is depicted in Figure 5b, where we report the relative percentage increase(/decrease) of the volume of tweets in the current time interval, when compared to the previous time interval (i.e., $\frac{(\#NewTweets - \#OldTweets)}{\#OldTweets}$). In case we have data in a square, we put the percentage of the difference with the previous window, otherwise the square appears without any number. This view can quickly reveal neighborhoods, where the twitter activity is rapidly increasing(/decreasing), signifying, for example, an aggregation of people in a specific location of the city.

More specifically, in Figure 5b, we present how the activity changes at Milan while getting closer to the beginning of the soccer game in the San Siro Stadium. As the figure shows, the square in which San Siro Stadium is located and it's

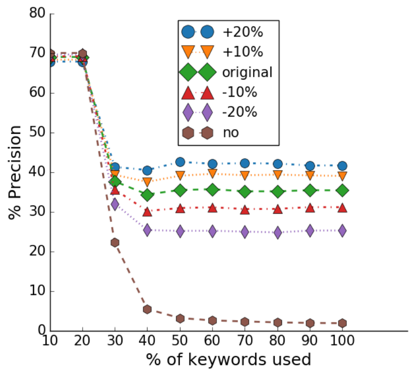


(a) Precision and Recall

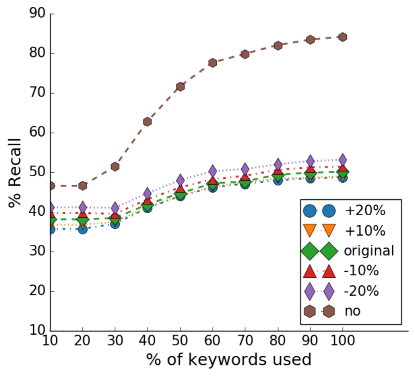


(b) F1

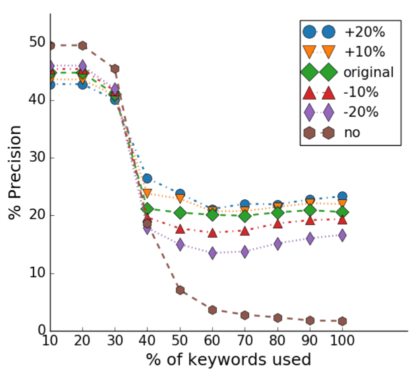
Fig. 2: Precision, Recall and F1 Comparison for 7 CGRs (@Top1)



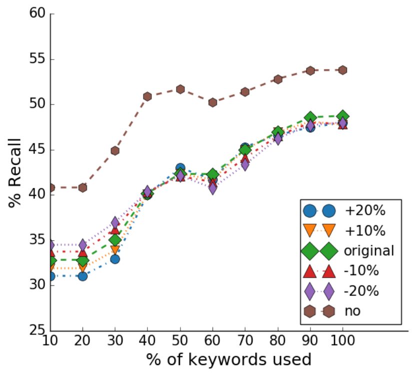
(a) Precision



(b) Recall



(c) Precision



(d) Recall

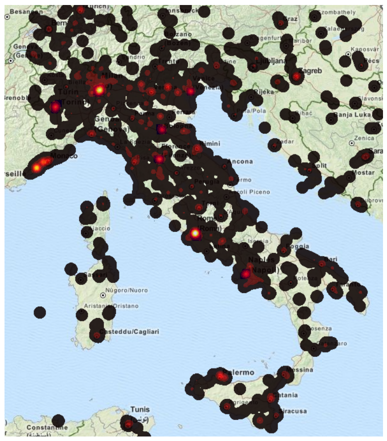
Fig. 3: (Top) Vatican (1.3km-side square / @Top1). (Bottom) San Siro (0.8km-side square / @Top1).

neighboring square have an increase of the activity, while the majority of the squares in Milan have a decrease. After further analyzing the activity of the squares, we identified that the square in which the San Siro Stadium is located has a constant increase of the activity up to 1 hour before the beginning of the match, while there is no other square with a constant increase. This representation allows us to depict the effect that such an event has on the area, and provides local authorities useful information that can be used to offer better services to the citizens (e.g., better manage the traffic flows).

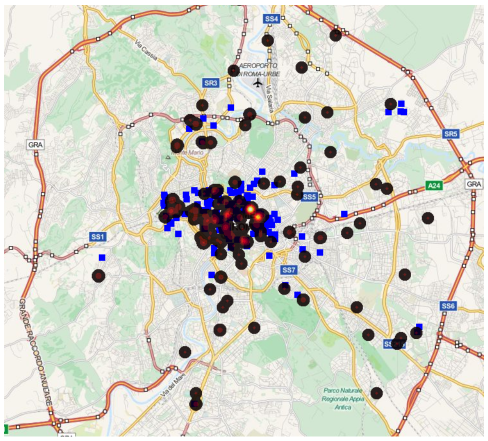
Finally, the user can focus on a specific location in the city, and view the volume of tweets over time for that location. Figure 6 illustrates the tweet activity time series[5] for the Vatican and San Siro locations (for the time interval of

[5] The labels of the peaks have been inserted manually, but could also been done automatically by analyzing the news streams.

May 26, 17.00 to June 1, 17.00, during which we had some important events). These graphs show that these two locations have very different characteristics: Vatican exhibits a relatively low, yet stable activity stream; on the other hand, San Siro has almost no activity for a large part of the time interval, but includes activity bursts. Nevertheless, it is interesting to note that, as reported in Figure 3, our algorithm for tweet geolocalisation performs equally well for both targeted locations.

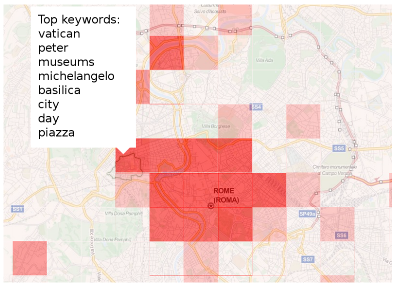


(a) Italy

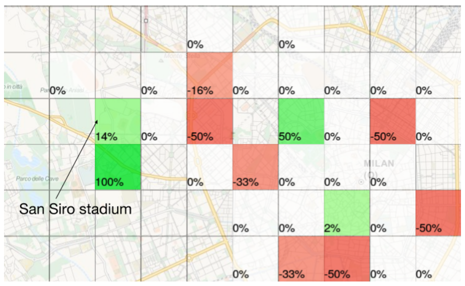


(b) Rome

Fig. 4: Heat-maps of geotagged tweets (data from one 4-hour window starting on June 2, 2016, at 10am).

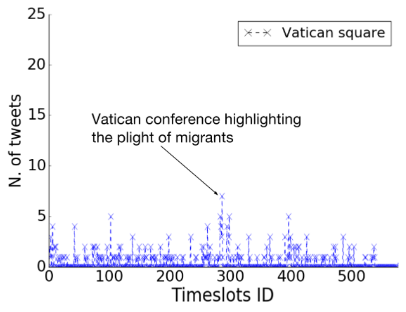


(a) Volume of Geotagged Tweets of Rome and Representative Keywords for Vatican square (June 2, 10:00-14:00).

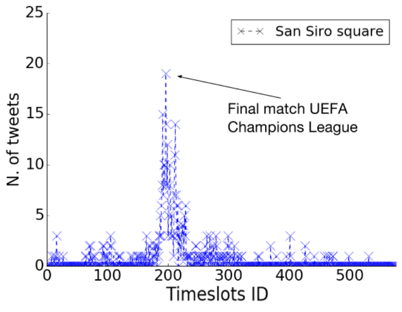


(b) Difference in the Activity of squares around San Siro after one slide (May 28, [15:00-19:00] to [15:30-19:30])

Fig. 5: Geotagged Tweets, Keywords and Activity Difference



(a) Vatican



(b) SanSiro Stadium

Fig. 6: Tweet volume time series.

7 Conclusions

In this work, we present a framework that allows us to geolocalise non-geotagged tweets. Our two-level framework allows the estimation of the location from which a post was generated, by exploiting the similarities in the content between this post and a set of geotagged tweets. Contrary to previous approaches, our framework provides geolocation estimates at a fine grain, thus, supporting a range of applications that require this detailed knowledge and could be used for social good. The experimental evaluation with real data demonstrates the effectiveness of our approach, regardless of the number (as small as 1) and size (as small as a 700m-side square) of the targeted locations. In order to render the results of our methods easier to understand, we developed a prototype dashboard that provides interactive visualizations, assisting end-users and large-scale event organizers to better plan and manage their activities.

Acknowledgments
This work was supported by a fellowship from Telecom Italia.

References

[1] Facebook,https://www.facebook.com/
[2] Google+,https://plus.google.com
[3] Twitter,https://twitter.com
[4] Abdelhaq, H., Sengstock, C., Gertz, M.: Eventweet: Online localized event detection from twitter. Proceedings of the VLDB Endowment 6(12) (2013)
[5] Ajao, O., Hong, J., Liu, W.: A survey of location inference techniques on twitter. Journal of Information Science 41(6), 855–864 (2015)
[6] Balduini, M., Bocconi, S., Bozzon, A., Della Valle, E., Huang, Y., Oosterman, J., Palpanas, T., Tsytsarau, M.: A case study of active, continuous and predictive social media analytics for smart city. In: ISWC Workshop on Semantics for Smarter Cities (S4SC)
[7] Balduini, M., Della Valle, E., DellAglio, D., Tsytsarau, M., Palpanas, T., Confalonieri, C.: Social listening of city scale events using the streaming linked data framework. In: ISWC (2013)
[8] Castillo, C.: Big Crisis Data: Social Media in Disasters and Time-Critical Situations. Cambridge University Press (2016)
[9] Chang, H.w., Lee, D., Eltaher, M., Lee, J.: @ phillies tweeting from philly? predicting twitter user locations with spatial word usage. In: ASONAM (2012)

[10] Cheng, Z., Caverlee, J., Lee, K.: You are where you tweet: a content-based approach to geo-locating twitter users. In: CIKM (2010)
[11] Crooks, A., Croitoru, A., Stefanidis, A., Radzikowski, J.: # earthquake: Twitter as a distributed sensor system. Transactions in GIS 17(1), 124–147 (2013)
[12] Earle, P.S., Bowden, D.C., Guy, M.: Twitter earthquake detection: earthquake monitoring in a social world. Annals of Geophysics 54(6) (2012)
[13] Eisenstein, J., O'Connor, B., Smith, N.A., Xing, E.P.: A latent variable model for geographic lexical variation. In: EMNLP (2010)
[14] Frias-Martinez, V., Soto, V., Hohwald, H., Frias-Martinez, E.: Characterizing urban landscapes using geolocated tweets. In: SocialCom-PASSAT (2012)
[15] Han, B., Cook, P., Baldwin, T.: Text-based twitter user geolocation prediction. JAIR (2014)
[16] Ikawa, Y., Enoki, M., Tatsubori, M.: Location inference using microblog messages. In: Proceedings of the 21st international conference companion on World Wide Web. pp. 687–690. ACM (2012)
[17] Johnson, S.: The Ghost Map: The Story of London's Most Terrifying Epidemic and How it Changed Science, Cities and the Modern World. Riverhead Books (2006)
[18] Kinsella, S., Murdock, V., O'Hare, N.: I'm eating a sandwich in glasgow: modeling locations with tweets. In: SMUC (2011)
[19] Leetaru, K., Wang, S., Cao, G., Padmanabhan, A., Shook, E.: Mapping the global twitter heartbeat: The geography of twitter. First Monday 18(5) (2013)
[20] Mathioudakis, M., Koudas, N.: Twittermonitor: trend detection over the twitter stream. In: SIGMOD (2010)
[21] Murdock, V.: Your mileage may vary: on the limits of social media. SIGSPATIAL Special (2011)
[22] Paradesi, S.M.: Geotagging tweets using their content. In: FLAIRS Conference (2011)
[23] Paraskevopoulos, P., Dinh, T.C., Dashdorj, Z., Palpanas, T., Serafini, L.: Identification and characterization of human behavior patterns from mobile phone data. NetMob (2013)
[24] Paraskevopoulos, P., Palpanas, T.: Fine-grained geolocalisation of non-geotagged tweets. In: Proceedings of the 2015 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining 2015. pp. 105–112. ACM (2015)
[25] Schulz, A., Hadjakos, A., Paulheim, H., Nachtwey, J., Mühlhäuser, M.: A multi-indicator approach for geolocalization of tweets. In: ICWSM (2013)
[26] Serdyukov, P., Murdock, V., Van Zwol, R.: Placing flickr photos on a map. In: SIGIR (2009)
[27] Sloan, L., Morgan, J.: Who tweets with their location? understanding the relationship between demographic characteristics and the use of geoservices and geotagging on twitter. PloS one 10(11), e0142209 (2015)
[28] Tsytsarau, M., Amer-Yahia, S., Palpanas, T.: Efficient sentiment correlation for large-scale demographics. In: SIGMOD (2013)
[29] Tsytsarau, M., Palpanas, T.: Survey on mining subjective data on the web. Data Mining and Knowledge Discovery (2012)
[30] Tsytsarau, M., Palpanas, T.: Nia: System for news impact analytics. KDD Workshop on Interactive Data Exploration and Analytics (IDEA) (2014)
[31] Tsytsarau, M., Palpanas, T., Castellanos, M.: Dynamics of news events and social media reaction. In: SIGKDD (2014)
[32] Tsytsarau, M., Palpanas, T., Denecke, K.: Scalable discovery of contradictions on the web. In: WWW (2010)
[33] Tsytsarau, M., Palpanas, T., Denecke, K.: Scalable detection of sentiment-based contradictions. DiversiWeb, WWW (2011)
[34] Yuan, Q., Cong, G., Ma, Z., Sun, A., Thalmann, N.M.: Who, where, when and what: discover spatio-temporal topics for twitter users. In: SIGKDD (2013)
[35] Zafarani, R., Liu, H.: Evaluation without ground truth in social media research. Communications of the ACM 58(6), 54–60 (2015)