TweeLoc: A System for Geolocalizing Tweets at Fine-Grain

Pavlos Paraskevopoulos
George Mason University
LIPADE, Paris Descartes University
pparaske@gmu.edu

Giovanni Pellegrini
University of Trento
giovanni.pellegrini@studenti.unitn.it

Themis Palpanas
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 identifying 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. Geolocalized information can be used by a variety of applications in order to offer better, or new services. However, only a small percentage of the twitter posts are geotagged, which restricts the applicability of location-based applications. In this work, we describe TweeLoc, our prototype system for geolocalizing tweets that are not geotagged, which can effectively estimate the tweet location at the level of a city neighborhood. TweeLoc employs a dashboard that visualizes the social activity of the geographic regions specified by the user, and provides relevant easy-to-access statistics. Moreover, it displays information on the way that these statistics evolve over time. Our system can help end-users and large-scale event organizers to better plan and manage their activities, and can complete this task fast and more accurately than alternative solutions that we compare to.**

Keywords: geotag, geolocation, Twitter, social networks, visualization

I. Introduction

[Motivation:] Every day activities and events affect people's lives, and in turn have a great impact on their social activity. The development of social networks such as Twitter, Facebook and Google+, allow their users to share whatever they see, do, or observe. The social interactions supported via these social networks have as a result the creation of information, the analysis of which could allow us to recreate (part of) the real-world. Furthermore, the development of mobile devices and the usage of the social networks via these devices provides to the user the possibility to share news and information in real-time, providing also the geographic location from which the post was made.

As a result, we now have access to datasets containing detailed information of social activities. To that effect, several applications and techniques 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. Applications use these datasets in order to characterize an urban landscape and optimize urban planning [1], to monitor and track mobility and traffic [2], [3], to identify and report natural disasters [4], or for analyzing the impact of events [5]. Therefore, such applications rely on the quality and quantity of data that include geolocalization information.

[Problem Description and Solution:] Although the use and analysis of geotagged posts is very appealing, only a very small percentage (around 2%) is geotagged [6], providing the exact location of the observation that is described in the post. The TweeLoc System addresses exactly this problem: it geolocalizes the non-geotagged posts, enabling the applications that need this information to produce better quality results. Furthermore, it offers an interactive visualization interface, facilitating the understanding and analysis of social activity and its evolution over time.

The TweeLoc system is based on our previous work on geolocalization of non-geotagged posts [7] (in particular, it employs the TG-TI-CLR1 algorithm). Our focus is on *fine-grained* location prediction: we wish to estimate the location of a post at the level of a city neighborhood. This is in contrast to previous approaches, which were predicting the geolocation of tweets at the level of regions, cities or zip-codes [8], [9], [10]. In our case, when the granularity becomes fine, the search space of the algorithms increases significantly. Nevertheless, the algorithms need to maintain a very high accuracy and, at the same time, be able to operate efficiently in a streaming fashion.

TweeLoc, provides interactive visualizations that include heatmaps for the depiction of the volume of (geotagged and geolocalized) tweets, and allows the user to zoom at different levels of granularity, ranging from a country, down to a city neighborhood. At the city neighborhood level, the user can also visualize the keywords that characterize that neighborhood. Finally, TweeLoc provides visualizations that illustrate in a comprehensive manner the changes in the volume of posts over time, for each neighborhood in a city (at a short time scale), as well as for an individual neighborhood (over long time intervals).

II. Related Work

Several works have presented and studied a range of problems of the identification of a geolocalization based on posts that are already geolocalized. Some works that study geolocalization issues rely on the similarity of user profile and location profile while some other build location profiles and try to match unique tweets with these locations.

Two representative works that belong to the first category are the studies presented in [11] and [8]. In the first study Cheng et al. propose the creation of location profiles based on idiomatic keywords and unique phrases mentioned in the tweets of users who have declared those locations as their origins, while in the second the authors create user profiles for the active users, and extract the keywords that are characteristic of specific locations (i.e., they usually appear in some location, and not in the rest of locations). For the extraction of these keywords they initially assign weights, and then prune them using a predefined keyword-weight threshold. This leads to a set of representative keywords for each location, which allows the algorithm to compute the probability that a given user comes from that location (Geometric-Localness (GL) method). A recent study evaluates the GL method, and compares it to other methods that solve the same problem: the experimental evaluation shows that the GL method achieves the best results [6]. Two additional studies target to geotag unique tweets [12], [13]. These two methods create chains of words that represent a location by using Latent Dirichlet Allocation (LDA) [14]. The latter study also takes into consideration the location a user has recorded as their home location.

A study that belongs to both categories, targeting to predict both a user's location and the place a tweet was generated from is presented in [9]. In this study, the authors construct language models by using Bayesian inversion, achieving good results for the country and state level identification tasks.

Even though the studies of the second group are closely related to our work, we observe that they operate at a much coarser time and space scale (e.g., space granularity of cities, or zip codes [9]). Moreover, previous studies rely on coarse-grain timeslots, and on the assumption of high volume training data being available at every timeslot (thus, leading to long-duration timeslots, in the order of weeks). On the contrary, in our work we predict the location of individual tweets at the granularity of city neighborhoods (in this study, we define a neighborhood as a square of 1km side).

The interested reader may also refer to a recent survey that discusses methods relevant to location inference [10].

III. Method Description

We now briefly describe the inner-workings of TweeLoc, and the TG-TI-CLR1 algorithm, which is language agnostic (for a detailed description, see our previous work [7]).

We first have to extract the most important keywords describing a particular location and its current activity. That is, we have to retrieve the geotagged tweets deriving from this location, create a signature keyword-vector, and find the similarity of the non-geotagged posts with this vector.

Extract Location Keyword-Vector: Initially, we gather all the geotagged tweets posted at a specific period of time (in our setup, a window of 4 hours), from each Coarse-Grained Location (CGL: in our setup, the different cities of a country) that we are interested in, and we group them into a single document for each CGL. Then, we compute the concordance of each word in each document, and we also employ the Tf-Idf model: $Idf_{keyword} = \log(\frac{n}{k})$, where n is the total number of documents, k is the number of documents that contain the specific keyword, and $Tf-Idf_{i,keyword} = \frac{count}{l} * Idf_{keyword}$, where l is the total number of keywords in document i. The use of the Tf-Idf model allows us to assess the importance of each keyword. Subsequently, we sort the keywords of each location according to their importance (i.e., we consider the $Tf - Idf$ score as the measure of the importance of each keyword), and we remove non-important keywords (i.e., common keywords, or stopwords are expected to have low $Tf - Idf$ scores). At the end, we obtain a keyword vector, *CGL-kv* representing each CGL. We repeat the same process for the Fine-Grained Locations (FGLs).

Location Activity Parameter: In addition to the analysis above, we examine the FGL activity volume, and compare that to the activity volume of the corresponding CGL. The intuition is that in the case of an important event, the number of posts in that FGL will be significantly increased, to the point that it will influence in the same way the number of posts in the CGL (where the FGL is located in). In order to capture this relationship, we measure the Pearson correlation between the two time-series (i.e., number of posts over time for CGL and FGL):

$$corr_{c,f} = \frac{\Sigma_{t=t_1}^{t_2}(Cts_t - \bar{Cts})(FLts_t - \bar{Fts})}{\sqrt{\Sigma_{t=t_1}^{t_2}(Cts_t - \bar{Cts})^2 \Sigma_{t=t_1}^{t_2}(Fts_t - \bar{Fts})^2}} + 1, \quad (1)$$

where t_1 and t_2 are the beginning and end times of the time-window we are interested in, t is the timeslot under examination (inside the larger time-window), c is the CGL, f is the FGL and Cts, Fts their activity time series. We also add 1 in order to shift the correlation range from [-1,1] to [0,2], so that candidate locations that correspond to positive correlation receive a bonus (they get multiplied by a number in the range (1,2]), while those that correspond to a negative correlation get penalized (they get multiplied by a number in [0,1)). Finally, we note that the above correlation can only be exploited if the activity (number of posts) is increasing. Therefore, we check the slope of the time-series (λ_{ts}) of every possible sub-window with length $n/2$ (in our setup, a window of 2 hours):

$$\lambda_{ts} = \frac{\Sigma((x - \bar{x})(y - \bar{y}))}{\Sigma(x - \bar{x})^2 \Sigma(y - \bar{y})^2}. \quad (2)$$

Only if a location has at least one sub-window with positive slope, it is considered as a candidate FGL.

Post Geolocalization: When we want to geolocalize a non-geotagged tweet, $Qtweet$, we constuct its keyword vector, compute its similarity (we use cosine similarity) to the keyword vectors of all candidate locations, and pick the most similar one. Each CGL that has a non-zero similarity with the $Qtweet$ is a "candidate location", and is further split into finer-grain candidate locations, i.e., the FGLs (in our setup, squares of 1km side). At the FGL level, the keyword-vector similarity includes an additional multiplicative

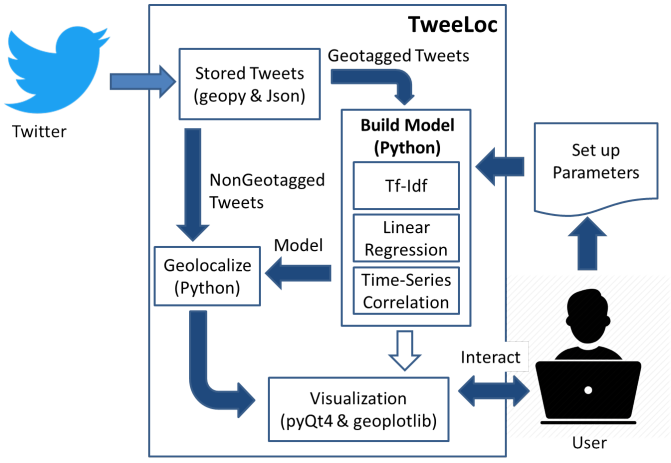


Fig. 1: TweeLoc Architecture

factor based on the correlation of the activity series. Finally, we sort the candidate FGLs according to their similarity, and those exceeding a dynamic threshold are considered as valid answers [7]. TweeLoc picks the FGL with the highest similarity score among those.

IV. The TweeLoc System

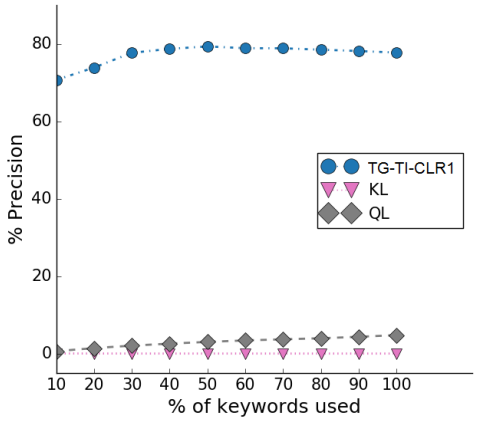
We now describe the TweeLoc architecture (see Figure 1).

The input to our system are the tweets deriving from the public API of Twitter, or alternatively from a $json$ file that contains historical tweets, as well as a file with all required initialization parameters. The parameters are user-defined, and they refer to the bounding box of the CGLs in interest, the space resolution of the FGLs (by default: 1 square km), the length of a timeslot in minutes (by default: 15), the number of timeslots in a window (by default: 16), the percentage of tweets to use for training (by default: 80%), the elasticity of the threshold (by default: +20%), whether we focus on a specific language or not (by default: no), the set of stopwords to filter out during preprocessing (by default: no stopwords filtered), and the percentage of keywords we want to keep in our keyword-vectors (by default: 60%).

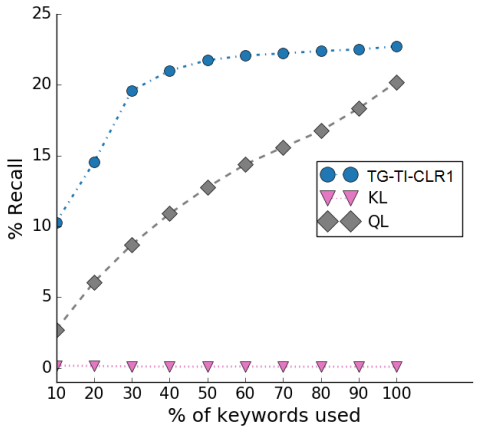
TweeLoc accesses the Twitter stream using the python library "geopy" [1]. The downloaded tweets are processed in batches (one timeslot at a time): initially stored in a $json$ file, and then "fed" to our system for building the model of each location (CGL and FGL). In this way, we can process both live and historical data using the same workflow. Note that the latency that this choice imposes to the processing of the live data (as low as a few minutes) is not a show-stopper for the applications targeted by TweeLoc.

The proposed system utilizes the TG-TI-CLR1 algorithm (described earlier) for building the model and estimating the locations of non-geotagged tweets. Previous studies [7] have shown that TG-TI-CLR1 is up to 3 times faster compared to the state-of-the art QL algorithm [9], while it achieves up

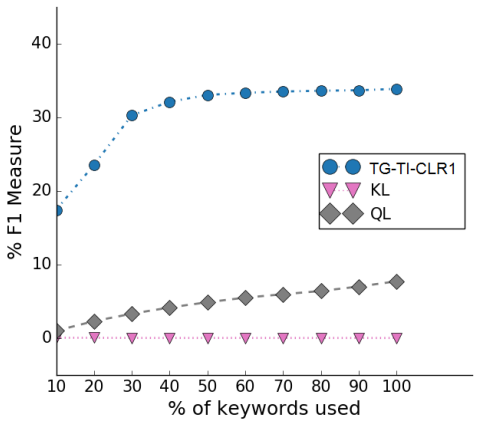
[1]https://github.com/geopy/geopy



(a) Precision Comparison



(b) Recall Comparison



(c) F1 Comparison

Fig. 2: Comparison with the state-of-the-art for 7 CGLs

to almost 4 times better F1, with recall more than 2 times better and precision up to 8 times better. The experimental results for precision, recall, and the F1 measure are shown in Figure 2. The difference in performance between TG-TI-CLR1 and QL can be explained by the different focus of

the QL algorithm, which was developed to operate at much bigger spatial (in the order of zipcodes, or cities) and temporal granularities (in the order of weeks, or months), relying on cases with high volume of training data. The decrease of the spatial and temporal granularity, has as a result the decrease of the volume of available information, which adversely affects the performance of the probabilistic models used by QL. In our figures we also include the results for another algorithm with similar characteristics to QL, the KL algorithm [9], which performs worse than QL. The detailed experimental results for TG-TI-CLR1 and QL, including the average execution times needed per window (4-hours window), are presented in Table I.

This part of the system was built using Python 2.7. The geolocalized tweets are then passed on to the visualization layer, which overlays their positions on maps, along with additional statistics.

The geographical maps that we use are composed of tiles downloaded from "Openstreetmap". These tiles change whenever we zoom-in or zoom-out. The visualizations that use heatmaps are using a modified version of the "geoplotlib"[2] Python library. Finally, we use the python library "pyQt4"[3] that handles graphic elements, and is useful for visualizing individual tweets on a geographical map, along with the tweet text and other metadata.

V. Demonstration

For the purposes of the demonstration[4], we will showcase a prototype of the TweeLoc system working with both static and live Twitter data. In what follows, we describe the datasets we will use, as well as the different ways the participants will be able to interact with the system. The goal is to demonstrate the benefits TweeLoc's fine-grained geolocalization, and its ability to support location-based applications that would otherwise not be possible. In order to showcase the significance of our contribution, we will also compare the results of TweeLoc with the results of the most prominent alternative solution, namely, the QL algorithm [9].

In the following paragraphs, we describe the datasets and scenarios we will use for the demonstration.

Datasets: The first dataset we will use contains Twitter geotagged posts that were generated in Italy between June 1 and June 20, 2016, which we will play back. The CGLs 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. We will also use a dataset from Germany, which contains 325,120 tweets, and a third dataset from the Netherlands, containing 232,454 tweets. The latter two datasets, were both generated between August 10 and September 11, 2014.

In addition, we will use live data from the Twitter public API, in order to demonstrate the real-time functionality of TweeLoc. The live data are going to be streaming tweets generated from USA during the days of the conference. We will also target tweets from New Orleans, where apart from ICDM, several other events will be taking place, such as Jazz and Blues music concerts.

1. Hotspot Identification: In our first demonstration scenario, the participants will experience how TweeLoc allows for a much more detailed spatial exploration of the data than previous methods. TweeLoc will first display to users a geographical map of the selected area, overlayed with a heatmap of all geolocalized tweets, as shown in Figure 3a (the black color corresponds to places with low activity, red with medium activity, and yellow with high activity). Unlike earlier approaches, the user will be able to zoom in a specific city in order to create a fine-resolution map (an example is shown in Figure 3b). At this level of detail, the user will observe the Twitter activity as it unfolds in the different neighborhoods of a city, and identify the most popular spots in the city.

In this scenario, the participants will be able to choose among the different datasets, and also interactively decide on which city (and for the case of the static datasets, the time interval, as well) to focus on.

2. Activity Analysis: In the second scenario, the users will concentrate on the analysis of the activity dynamics of the tweets. The interface depicted in Figure 4a visualizes a heatmap based on the number of tweets that were posted from each individual FGL (i.e., square in the grid). In this view, when the user hovers with the mouse over a square, a bubble appears that shows the representative keywords of that square, corresponding to the content of the tweets of that square. The user could also switch to an alternative view, visualizing a differential heatmap (Figure 4b), which visualizes the way that the activity of each FGL changes (i.e, increases, or decreases) between two timeslots. In this case, each square shows the percentage of the activity change, and is colored in green when the activity increases over time, otherwise in red. In all heatmap views, the upper right corner of the window displays the name of the heatmap, the starting time of the window, and its length in minutes. This scenario will demonstrate the ability of TweeLoc to reveal the activity of different neighborhoods in a city and identify hotspots, explain this activity in terms of the contents of the tweets, and also explore how this activity evolves over time.

In this scenario, the participants will be able to explore the Twitter activity dynamics for different cities, and also decide on the dataset used (including the live stream). They will also be able to navigate across time (except for the live dataset), effectively changing the time window (i.e., timeslot) under consideration.

3. Targeted Statistics: In our third demonstration scenario, the users will be able to check the location of specific, individual tweets, as they appear in the live stream. The text of the tweets will be displayed on the screen, and when clicked on, the system will display the predicted position of that tweet on the map, by automatically zooming-in to the FGL identified as the tweet location (Figure 5a). The system will additionally display a list of representative keywords for all the

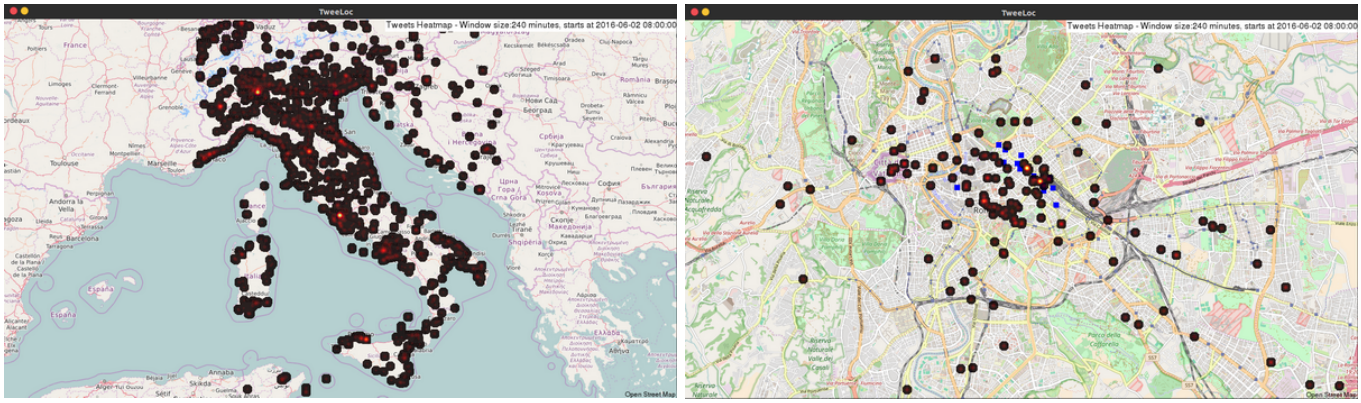
[2]https://github.com/andrea-cuttone/geoplotlib

[3]https://pypi.python.org/pypi/PyQt4

[4]Video available at: https://www.dropbox.com/sh/tjqaipfn71h9ubp/AADMKSd-EKDkezuzccXCtzJva?dl=0

	TG-TI-CLR1			QL		
Perc. of Keywords	Time (sec)	Precision	Recall	Time (sec)	Precision	Recall
10%	28	0.72	0.10	90	0.05	0.03
20%	32	0.76	0.14	90	0.06	0.06
30%	41	0.79	0.20	90	0.06	0.09
40%	50	0.79	0.22	90	0.06	0.10
50%	58	0.79	0.23	90	0.06	0.12
60%	69	0.79	0.23	90	0.06	0.14
70%	79	0.79	0.23	90	0.07	0.15
80%	88	0.78	0.23	90	0.07	0.16
90%	99	0.76	0.23	90	0.07	0.18
100%	123	0.75	0.24	90	0.07	0.20

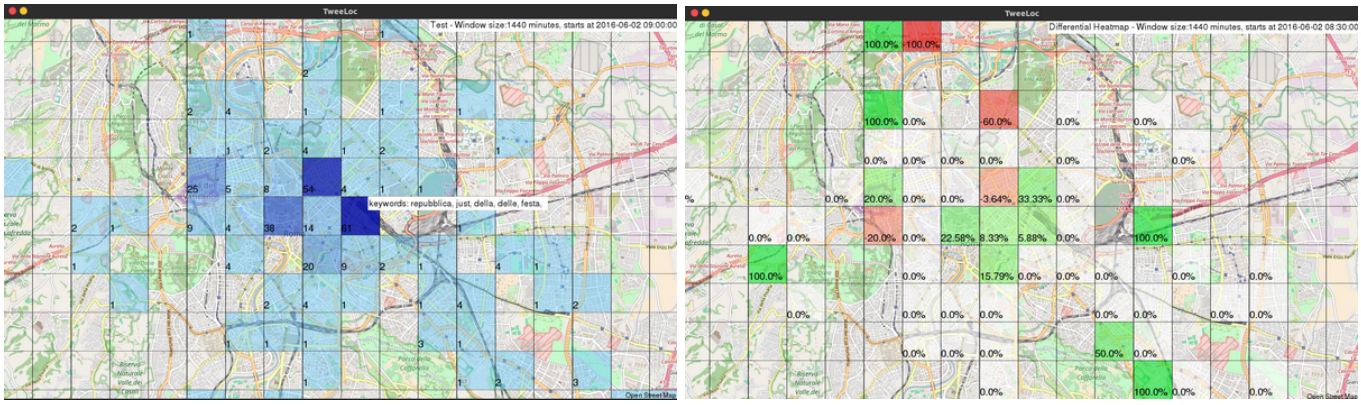
TABLE I: Average Execution Time per Window (in sec) and Performance Comparisons



(a) Country Activity Heatmap

(b) Rome Activity Heatmap

Fig. 3: Country and City Activity Heatmaps



(a) FGL Activity

(b) Differential Heatmap

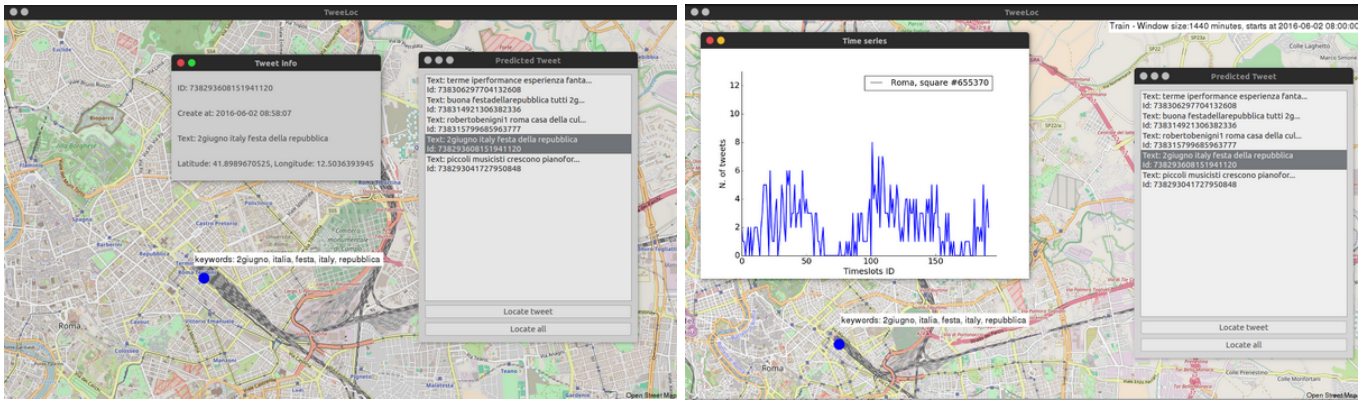
Fig. 4: FGL Activity and Differential Heatmaps

tweets posted from that same FGL. Furthermore, by clicking on the FGL position, a new window will pop-up, depicting the volume of tweets over time that were posted from that FGL (Figure 5b). The interface will also provide a "Locate all" button, for geolocalizing all the individual tweet posts currently displayed on the screen.

In this scenario, the participants will be able to choose individual tweets from the live stream (in case of a network problem, we will play back one of the recorded datasets).

4. Comparison to State-of-the-Art Solution: In the last demonstration scenario, the users will have the chance to compare the performance of TweeLoc to that of the QL algorithm [9], which is the state-of-the-art solution. In particular, the users will be able to employ QL for the Targeted Statistics scenario, and compare its performance both visually and analytically to that of the TG-TI-CLR1 algorithm: the system will display on the map the geolocations predicted by each one of the algorithms, along with the true geolocation, as well as the cummulative distance (error) for each algorithm.

In this case and for the purposes of the comparison, the



(a) Check Tweet Details Interface

(b) FGL Activity

Fig. 5: FGL and Activity Tweet Details

participants will choose individual tweets from one of the recorded datasets, for which the ground truth (i.e., the true geolocation) is known.

VI. Conclusions

In this work, we present TweeLoc, a system that can effectively and efficiently geolocalize non-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. Our system provides a variety of visualizations and statistics, which enable users and analysts to quickly understand how social activity evolves over space and time.

Acknowledgments

This work was supported by a fellowship from Telecom Italia.

References

[1] V. Frias-Martinez, V. Soto, H. Hohwald, and E. Frias-Martinez, "Characterizing urban landscapes using geolocated tweets," in *SocialCom-PASSAT*, 2012.

[2] M. Balduini, E. Della Valle, D. DellAglio, M. Tsytsarau, T. Palpanas, and C. Confalonieri, "Social listening of city scale events using the streaming linked data framework," in *ISWC*, 2013.

[3] P. Paraskevopoulos and T. Palpanas, "What do Geotagged Tweets Reveal about Mobility Behavior?" in *Mobility Analytics for Spatio-temporal and Social Data (MATES), in conjunction with the International conference on Very Large Data Bases (VLDB)*, 2017.

[4] A. Crooks, A. Croitoru, A. Stefanidis, and J. Radzikowski, "# earthquake: Twitter as a distributed sensor system," *Transactions in GIS*, vol. 17, no. 1, pp. 124–147, 2013.

[5] P. Paraskevopoulos, G. Pellegrini, and T. Palpanas, "When a tweet finds its place: fine-grained tweet geolocalisation," in *International workshop on data science for social good (SoGood), in conjunction with the European conference on machine learning and principles and practice of knowledge discovery (ECML PKDD)*, 2016.

[6] B. Han, P. Cook, and T. Baldwin, "Text-based twitter user geolocation prediction," *JAIR*, 2014.

[7] P. Paraskevopoulos and T. Palpanas, "Where has this tweet come from? fast and fine-grained geolocalization of non-geotagged tweets," *Social Network Analysis and Mining*, vol. 6, no. 1, p. 89, 2016.

[8] H.-w. Chang, D. Lee, M. Eltaher, and J. Lee, "@ phillies tweeting from philly? predicting twitter user locations with spatial word usage," in *ASONAM*, 2012.

[9] S. Kinsella, V. Murdock, and N. O'Hare, "I'm eating a sandwich in glasgow: modeling locations with tweets," in *SMUC*, 2011.

[10] O. Ajao, J. Hong, and W. Liu, "A survey of location inference techniques on twitter," *Journal of Information Science*, vol. 41, no. 6, pp. 855–864, 2015.

[11] Z. Cheng, J. Caverlee, and K. Lee, "You are where you tweet: a content-based approach to geo-locating twitter users," in *CIKM*, 2010.

[12] J. Eisenstein, B. O'Connor, N. A. Smith, and E. P. Xing, "A latent variable model for geographic lexical variation," in *EMNLP*, 2010.

[13] S. M. Paradesi, "Geotagging tweets using their content." in *FLAIRS Conference*, 2011.

[14] D. M. Blei, A. Y. Ng, and M. I. Jordan, "Latent dirichlet allocation," *the Journal of machine Learning research*, 2003.