P-GENT: Privacy-Preserving Geocoding of Non-Geotagged Tweets

1st Shuo Wang  
Computing and Information Systems  
University of Melbourne  
Melbourne, Australia  
shuow4@student.unimelb.edu.au

2nd Richard Sinnott  
Computing and Information Systems  
University of Melbourne  
Melbourne, Australia  /rsinnott@unimelb.edu.au

3rd Surya Nepal  
Data61  
CSIRO  
Sydney, Australia  
Surya.Nepal@data61.csiro.au

Abstract—With the widespread proliferation of location-aware devices and social media applications, more and more people share information on location-based social networks such as Twitter. Such data can be beneficial to better plan and manage individual’s activities and other social applications, e.g., location-based advertisement or recommendation. However, only a very small proportion of tweets are geotagged due to privacy concerns or lack of underlying positioning infrastructures. Hence it is meaningful to estimate the geographic information for non-geotagged tweets, i.e., geocoding, which can help to improve the applicability and utility of social media data. Contrary to existing geocoding approaches, this paper aims at the privacy risk and providing a fine-grained estimation. In this paper, we propose Privacy-preserving GEocoding of Non-geotagged Tweets (P-GENT) for geocoding non-geotagged tweets with fine-grained estimation whilst protecting privacy. Our approach estimates the geographic location of a non-geotagged tweet based on the similarities between the content of the tweet and the keyword lists of detected local events from the archived geo-tagged tweets during the same time period. This approach implements a spatio-temporal clustering algorithm to discover local events with a fine-grained granularity and an important keyword extraction mechanism to describe the detected local event. In addition, a density-seed discovery approach is used to reduce the sparseness of geo-tagged tweets and the time complexity of clustering approach. The experimental evaluation with real-world data demonstrates that our approach has at most 92% precision for one timeslot and 33% precision remained for all time slots after using privacy-preserving mechanisms.

Index Terms—Differential privacy, location estimation, spatio-temporal clustering, event detection

I. INTRODUCTION

The last decade has witnessed the widespread proliferation of social networks, e.g., Twitter, and Google+. People share information on a wide variety of real-world activities by social network applications, ranging from widely known events (FIFA World Cup or political elections) to local ones (open-air assembly in a park). With the rapid development of mobile and location-aware devices, activities can be logged in real time and embedded geographic information. Various applications and techniques benefit from social networks data, especially geotagged data, to better plan and manage individual activities as well as commerce, urban management and scientists alike. The more geotagged data are available, the higher the probability of being able to produce better quality results of these applications and techniques [1]. However, only a tiny amount of tweets are geotagged mainly due to privacy concerns. For Twitter, the proportion of geotagged tweets is a mere 1.5%−3% due to privacy concerns or lack of underlying positioning infrastructures, which restricts the utility of the applicability and utility of many social and business opportunities.

Existing works have addressed this problem by geocoding non-geotagged tweets, i.e., location inference methodologies and algorithms, such as Natural Language Processing (NLP) technique-based [2], Gazetteers-based [3], and probabilistic and machine learning technique-based [4], [5] approaches. Diverse solutions have achieved varying effectiveness and granularity levels. Some of them achieve a coarse-grained inference of the location of non-geotagged tweets, e.g., the level of postal zip codes, cities or states. A fine-grained granularity, e.g., geographic coordinate level, can enable a new range of applications that need finer granularity data. Some existing works apply a regular geodesic grid granularity to divided the Earth’s surface into discrete rectangular cells (regions of 1° of latitude by 1° of longitude), and conduct location inference using cells. Such cell or grid-based approaches are time and memory consuming due to the sparseness issues, i.e., posts concentrate around popular regions and a lot of cells associate to few textual contents. The location entity resources used in gazetteers-based are always some popular landmarks, which is hard to scale for rural or residential areas, as well, there are a lot of tweets go beyond the explicit geocoding or place name mentions. Further, the inference of location (at any granularities) for non-geotagged tweets may compromise individuals’ privacy (e.g., home location, political views or categories of disease based on their visited locations). The risk of revealing individual privacy are exacerbated when the adversaries possess various background knowledge. It is essential that personal and sensitive information is not leaked when estimating the location of non-geotagged tweets. In this paper, we address these problems by proposing a framework for inferring the location of individuals’ non-geotagged tweets at the level of geographic coordinates, i.e., geocoding non-geotagged tweets, with privacy protected under a robust privacy definition. Our approach estimates the geographic location of a non-geotagged tweet based on the similarities between the content of the post and a local event set derived from archived geotagged tweets.
In this work, we propose a framework to estimate the fine-grained location of non-geotagged tweets using the similarities between the content of a non-geotagged tweet and a local event set derived from the training set of geotagged tweets. The contributions we make in this framework can be summarized as follows.

To handle the sparseness and time complexity issues, we use an automatic cluster center discovery approach to identify density-seeds from a large-scale collection of geotagged tweets data to explore the spatial distribution of geotags in a domain. In addition, we apply density-seeds to generate the initial clustering results for the following spatio-temporal clustering procedure to speed up the runtime of clustering.

To maintain fine granularity of local event detection, we propose an OSP-Cluster spatio-temporal clustering to detect local events from geo-tagged tweets. The spatio-temporal clustering method is used to identify the place where and the time when for tweets that are intensively posted and potentially referred to the same event. Then we identify the co-occurrences of keywords in each spatio-temporal cluster (OSP-cluster) to detect the occurrences of local events in terms of both spatial and temporal dimensions. Further, spatial outliers, appearing when keywords exhibit a non-local behavior and are used far away from the location of an event, are eliminated by a filtering mechanism.

To achieve a fine-grained location estimation of non-geotagged tweets, we use an event matching model based on three similarity quantities of tweets with local events for geocoding. The goal of the event matching model is to match a non-geotagged tweet to a local event, if possible, where the matching objects are a list of local events in a given domain at a certain time slot. This matching model utilizes the content similarity weighted by TF-IDF values, spatial proximity between user and event’s locations as well as the popularity of event to estimate the probability of the given tweet that was posted at the location of a local event.

In addition, to address the privacy concerns, we embed robust privacy-preserving mechanisms under differential privacy to sanitize the estimated results for non-geotagged tweets before releasing for public use. Finally, we perform a detailed experimental evaluation of our approach, using real data from Twitter. The results demonstrate the efficiency and effectiveness of the proposed approach when compared to various alternatives.

The rest of this paper is structured as follows. Background and problem formulation is presented in section 2. Section 3 describes the ideas and mechanisms of the P-G ENT approach. Section 4 presents the evaluation metrics and the experimental performance of the approach. A survey of related work is given in section 5. Section 6 draws conclusions on the work as a whole.

II. PROBLEM FOUNDATION

A. Problem Definition

The basic goal of this article is to estimate the geographical location of the non-geotagged tweet. We divided each day into six time slots according to human’s daily activities, i.e., $TS=\{(6:00-10:00),(10:00-14:00),(14:00-18:00),(18:00-22:00),(22:00-2:00),(2:00-6:00)\}$. The time interval we set as the duration of 4 hours, which can effectively capture an important local event. Then we detect local events using spatio-temporal clustering approach with a smaller temporal duration, e.g., every 15 minutes interval. More formally, given a non-geotagged tweet $ng\in NG : ts \in TS$ at a given time slot $ts=[t_x,t_y]$, and a set of archived geotagged tweets $GT=\{tw_{l_1}^u,\ldots,tw_{l_i}^u,\ldots,tw_{l_k}^u\}$ at the given time slot $ts$ in a certain domain, where $l_i$ is the location (or density-seed) where the tweet was generated at $t_j$, we aims to identify the location $l_x$ where $ng\in ts$ was generated. We focus on fine-grained location predictions, i.e., at the level of a density-seed (which is usually similar as a POI or a cluster of POIs).

Such estimation problem can be defined as a probabilistic model that evaluates the similarity between a non-geotagged tweet and a local event. Let $U$ be a set of users, where $u \in U$ has a location $l(u)$. Let $E$ be a set of local events (localized at spatio-temporal clusters), where $e \in E$ has a set of attributes including the geolocation $l(e)$ (e.g., the centroid of a spatio-temporal cluster). In addition, let $NG$ be a set of non-geotagged tweets, where $ng \in NG$ consists of the context of the tweet, the associated user $u_{ng} \in U$, and needs to be matched to a local event $e \in E$. We aim to model a probability distribution $P$ over $NG \times U \times E$. Then the $P(e,ng,u)$ is calculated for a triple $(e,ng,u)$ drawn from $P$, so as to estimate the best $e^*$ for an non-geotagged tweet-user pair $(ng,u)$.

We aim to decide the probabilistic model, which can be expressed as $P(e,ng,u) = P(e,u)P(ng|e,u)$. There are two components of the probabilistic model. (1) The spatial similarity component that captures the relationship between $U$ and $E$, given by $P(e,u)$ to reveal the probability of a user $u$ appearing at the location of a local event $e$. (2) The semantic similarity component that captures the relationship between $E$ and $NG$, given by $P(ng|e,u)$ to describe the similarity between terms of a non-geotagged tweet $ng$ corresponding to a user $u$ and that of a local event $e$.

B. Preliminary Concepts

**Definition 1. (Location Entropy).** Given a location $l$, let the $GT_l$ be the set of geotags (visits) to location $l$ by all social media users. Let $s_l = |GT_l|$ represent the number of visits at location $l$. Let $U_l$ be the set of distinct individuals who visit at location $l$, and $GT_{l,u}$ be the set of visits that user $u$ has made at location $l$. The $s_{l,u} = |GT_{l,u}|$ is represented as the number of visits that user $u$ made to location $l$. Thus $p_{l,u} = \frac{|GT_{l,u}|}{|C_l|}$ denotes the portion of total visits that belongs to user $u$. The location entropy of $l$ based Shannon entropy [6] is denoted by:

$$H(l) = H(p_{l,u1}, p_{l,u2}, \ldots, p_{l,u|U_l|}) = - \sum_{u \in U_l} p_{l,u} \log_2 p_{l,u}$$

(1)
A higher location entropy value means the visits on this location are more evenly distributed among users that visited this location, i.e., it is a more popular location.

**Definition 2.** \((α\text{-dataset})\). The \(α\text{-dataset}\) for a given non-geotagged tweet \(ng\), denoted by \(α_{ng}\text{-dataset}\), is composed of a set of possible geolocations or centroids of spatio-temporal clusters where the tweet might be generated. Given the estimated probability of each candidate, \(α\text{-dataset}\) for \(ng\) is a set of candidates consisting of the top \(k\) most possible candidates with cumulative probability more than \(1 - α\). Finally, a possible candidate from \(α_{ng}\) is selected to be the geographic reference for \(ng\), which can be the most possible candidate or randomly-chosen one under privacy-preserving mechanisms.

**C. \(ε\)-Differential privacy**

**Definition 3.** \((ε\text{-Differential privacy [7]})\). Let \(ε > 0\) be a constant, a randomized function \(F\) satisfies \(ε\)-differential privacy if for all datasets \(D1\) and \(D2\), differing at most by one element from each other, all outcomes of the database \(S\) \((S \subseteq \text{Range}(F))\) there is:

\[
Pr[F(D_1) \in S] \leq e^\epsilon Pr[F(D_2) \in S]
\]  

(2)

The parameter \(ε > 0\) is called the privacy budget and it allows users to control the level of privacy. A smaller privacy budget suggests more limits posed on the influence of an individual item, leading to a stronger privacy protection. The standard mechanism to achieve differential privacy is the sensitivity method [7], [8] (e.g., Laplacian mechanism or Exponential mechanism) that adds the noise to the query output according to the sensitivity of the query function. The sensitivity reveals the maximum change in the query answers due to the change of a single database entry.

**Definition 4.** \((\text{Sensitivity [7]})\): The sensitivity of a query function \(f\) is the maximum change in the query results:

\[
\Delta(f) = \max_{D_1,D_2} |f(D_1) - f(D_2)|
\]  

(3)

**III. Proposed Approach**

In this section, we describe the P-GENT approach in details. Generally, the P-GENT framework consists of three components: density-seed discovery (C1), local event detection and keyword extraction (C2), and event matching-based location inference and perturbation (C3). The overview of the P-GENT approach is illustrated in Fig.1.

C1 (Step 1) is used to handle the sparseness and complexity issues. An automatic cluster center discovery approach is adopted to identify density-seeds from a large-scale collection of geotagged tweets data, corresponding to finer spatial distribution (in this case, they consist of the centers of location clusters). In addition, we apply density-seeds to generate the initial clustering results for the following spatio-temporal clustering procedure to speeds up the runtime of clustering. The output of C1 is a density-seed set.

C2 is proposed to detect local events using the density-seeds from C1 and geotagged tweets at the same time slot as the given non-geotagged tweet. It can be divided into spatio-temporal clustering (Step 2) and local event detection and keyword extraction (Step 3). The spatio-temporal clustering method is used to identify the place where and the time when for tweets that are intensively posted and then potentially referred to the same event. Then we identify the co-occurrences of keywords in each spatio-temporal cluster to detect the occurrences of local events in term of both spatial and temporal dimensions. Further, spatial outliers, appearing when a keywords exhibit a non-local behavior when used far away from event location, are eliminated by a filtering mechanism. The output of C2 is a local event set LEc, associated OSP-clusters.

C3 is applied to estimate the possible geolocations of a given non-geotagged tweet, consisting of similarity evaluation, event matching (Step 4) and perturbation mechanism for estimation results (Step 5). The event matching model is based on three similarity quantities of tweets with local events for geocoding, i.e., the interest of the user to the location of local event derived from the spatial proximity between them, the similarity of content between the given non-geotagged tweet and the keyword vectors of local events weighted with TF-IDF values, and the popularity of the location where the local event occurs. The goal of the event matching model is to match a non-geotagged tweet to a local event, if possible, where the candidates are a list of local events in a given domain at the same time slot with the given post. Then, the probability of the given tweet that was generated at the location of a local event is identified. Further, to address the privacy concerns, we embed robust privacy-preserving mechanisms under differential privacy to sanitize the estimated results for non-geotagged tweets before releasing for public use.

These three components are elaborated in the following sections.

**A. Density-seed Discovery-C1**

The density-based clustering algorithms can be applied to identify the dense locations that with higher local density and separation from other locations generally, i.e., density-seeds, such as density-based spatial clustering of applications with noise (DBSCAN) [9], ordering points to identify the clustering structure (OPTICS) [10], mean-shift clustering (MSC)
The CFSFDP algorithm provides a fast clustering solution by finding density peaks in the dataset, consists of discovering cluster center (density peak) with higher local density and separation from other cluster centers, and generating cluster by calibrating each remaining point to its nearest neighbor with higher density to be the same cluster. Consequently, we can use the cluster center identification mechanism of CFSFDP as one-step density-seed discovery approach. In addition, the CFSFDP algorithm not only considers the connectivity but also the separation of points to construct clusters. Hence it is robust with respect to the radius of a neighborhood, compared to other density-based algorithms. There are two assumptions in CFSFDP algorithm [12]: (1) cluster center is the highly dense point that is surrounded by neighbors with lower local density; (2) cluster center lies at relatively large distances from points with higher local density (or other cluster centers). Based on the density peak detection solution, the density-seed discover approach is illustrated as follows.

For each data-point \( p_i \), there are two quantities need to be evaluated: local density \( \rho(p_i) \) and separation \( \delta(p_i) \) from nearest high local dense point. Given a distance measure \( dis(\cdot) \) that is assumed to satisfy the triangular inequality and a cutoff distance \( d_c \) to discover the \( d_c \)-Neighbourhood \( NB(p_i) = \{ p_j | dis(p_i, p_j) < d_c, p_j \in P \} \), i.e., the neighborhood of point \( p_i \) within the radius of \( d_c \). \( d_c \) can be chosen based on the heuristic approach that in average there exist 1 to 2 \% of neighbors in a dataset [12]. The density and separation measures are given as:

\[
\rho(p_i) = |NB(p_i)| \tag{4}
\]

\[
\delta(p_i) = \begin{cases} 
\min_{j: \rho(p_j) > \rho(p_i)} \{ dis(p_i, p_j) \}, & \text{if } \exists j : \rho(p_j) > \rho(p_i) \\
\max_{p_j \in P} \{ dis(p_i, p_j) \}, & \text{otherwise.} 
\end{cases} \tag{5}
\]

The local density measure \( \rho(p_i) \) is used to count the number of points in the \( d_c \)-Neighborhood of point \( p_i \). The separation measure \( \delta(p_i) \) is used to evaluate isolation of \( p_i \) from other points having higher density. \( \delta(p_i) \) is much larger for locally or globally high dense points, and the smaller the value of \( \delta(p_i) \), the greater is the possibility that \( p_i \) is calibrated to the same cluster as its nearest neighbor with higher density. Then a heuristic solution is used for the selection of expected cluster centers manually by plotting calculated values of local density \( \rho(p_i) \) and separation \( \delta(p_i) \), e.g., Fig. 2b. As shown in Fig. 2b, points with large \( \rho(p_i) \) and relatively large \( \delta(p_i) \) can be selected as cluster centers compared to other cluster points while the points with higher \( \rho(p_i) \) and low \( \delta(p_i) \) (points of black) are treated as halo clusters (noise or outliers). Then, cluster centers are used as density-seeds.

However, this center detection approach derived from ordinary CFSFDP needs to manually choose some points as cluster centers to represent the clusters based on the density and separation of each point. Such drawback limits the implementation for the automatic density-seed discovery due to the subjectivity and arbitrariness of cluster center selection. Hence, we propose an automatic density-seed discovery strategy to improve automaticity of density peak selection solution used in the ordinary CFSFDP.

The basic idea of the automatic density-seed discovery strategy is to evaluate centering degree of points based on the normalization cross of \( \rho(p_i) \) and \( \delta(p_i) \) then to select the points with the largest centering degrees as the density-seeds (cluster centers). The centering degree is defined as follows:

**Definition 5. (Centering Degree-\( \gamma \).)** Given the local density \( \rho(p_i) \) and separation \( \delta(p_i) \) for a point \( p_i \), the centering degree is deviation degree of a point to the origin point of the plots of \( \rho(p_i) \) and \( \delta(p_i) \), i.e., the probability of the point to be a cluster center. The formal definition is in Equation (6).

\[
\gamma(p_i) = \frac{\rho(p_i) \times \delta(p_i)}{\sum_{j} \rho(p_j) \times \delta(p_j)} \tag{6}
\]

To describe the selection of cluster centers, we choose the top \( m \) points with the largest centering degree values from the centering degree value list in decreasing order, generally \( m < |P|/3 \), as illustrated in Fig. 2c. As shown, the centering degree values have different degree drop trends, and the inflection point can be used as the threshold to detect centering degree values.

**Definition 6. (Inflection Point).** It is defined as the point where the centering degree values decreasing trend is the biggest in the plots of centering degree values, denoted by the \( x^{th} \) point of the top \( k \) point set sorted by centering degree values in decreasing order. The \( x \) value can be decided according to Equation (7).

\[
x = \arg\max \{ \theta(i_{0}, i, i + n) \} \tag{7}
\]

Here the \( \theta(i_{0}, i, i + n) \) is the rate of change of the second-order centering degree values, defined in Equation (8), to reflect the speed of the downtrend of \( \gamma \).

\[
\theta(i_{0}, i, i + n) = \frac{\text{slope}_{i+1}^{i+n}}{\text{slope}_{i_{0}}} \tag{8}
\]
\[ \text{slope}_{i+n} = \frac{\gamma(p_{i+n}) - \gamma(p_i)}{n} \quad (9) \]

In this case, we set \( i_0 = 1 \) and \( n = 1 \). Fig. 2d illustrates the \( \theta \) of the serial number of sorted candidate points, where the serial number with the maximum \( \theta \) can be used to decide the inflection point \( x \). Based on the selected \( \theta \), the cluster center set are generated as \( O = \{ o_u | u = 1, 2, \cdots, x \} \). Thus, we apply the density-seed discovery approach on the training geospatial database, as shown in Algorithm 1. Note that we choose the top \( m < |P|/3 \) point as the potential cluster centers candidates according to the \( \gamma \) values, since the number of cluster is much less than \( |P| \).

Algorithm 1: Automatic Density-Seed Discovery

Input: The set of locations \( P, d_c, m \).
Output: Density-Seed Set \( DSS \).
1) Figure out \( d_c := \) \( \max \{d_i \} \).
2) Use Equation(4) to calculate the \( \rho \) for each point and sort in descending order.
3) for \( i = 1 \) to \( |P| \) do
   4) Use Equation(5) to calculate \( \delta(p_i) \).
   5) \( \text{Gamma}[i] = \rho(p_i) \times \delta(p_i) \).
   6) \( \text{GammaSum} = \text{Gamma}[i] \).
   7) \( \text{Gamma}[i] = \text{Gamma}[i]/\text{GammaSum} \).
   8) \( \text{List} \left\langle \text{gamma}, \text{label} \right\rangle \leftarrow \text{Gamma}.\text{SortDecend} \).
   9) for \( i = 1 \) to \( m \) do
      10) Calculate \( \text{slope}_i \) and \( \text{slope}_{i+1} \);
      11) Calculate \( \theta(1, i, i+1) \);
      12) Inflection Point \( x = \argmax\{\theta(1, j, j)\} \);
      13) \( DSS = \{ o_u | u = 1, 2, \cdots, x \} = \text{List}.\text{Top}(x) \).

B. Local Event Detection and Keyword Extraction-C2

1) spatio-temporal Clustering: In order to detect local events, we perform a spatio-temporal clustering on geotagged tweets during a given time slot. The problems of spatio-temporal clustering are to handle the repetitive clustering procedures for different time slots each day and to match the scale of spatio-temporal clusters to the actual range of local events. Specifically, it is time and resource consuming to perform the spatio-temporal clustering on historical geotagged tweets database during every time slot at each day, and to detect co-occurrences of keywords of a local event if the local event related tweets are divided into several clusters or the cluster is larger than a local event due to irrelevant tweets. The commonly-adopted density-based clustering algorithms, e.g. DBSCAN, aggregate many locations within the density definition, and each of these can be used to discover further density-reachable locations. Areas with a high density of points are defined as clusters. The basic idea of the DBSCAN algorithm is that, for each point of a cluster, the neighborhood within \( \text{Eps} \) has to contain at least \( \text{MinPts} \) points, called \( \text{Eps} \)-neighborhood, i.e., the density in the neighborhood has to exceed the predefined threshold. However, it consumes massive resources (memory, storage) and can be computationally infeasible for big data scenarios, due to a large number of iterations of range queries and expanding procedures.

To tackle this problem, we use a variant from DBSCAN, called Omega-Cluster, combining the density-seed set. This approach uses the notion of a \( \Omega \)-combinable relation instead of the density-connected relation used in DBSCAN. Consequently, we extend Omega-Cluster clustering approach to an adaptive spatio-temporal clustering approach, called OSP-Cluster, that use the density-seeds to generate an initial clustering result set to reduce the number of iterations, since the local event always appears at locations where geotagged tweets have been generally frequently posted, e.g., popular landmarks, accident blackspots or home locations. This can adjust the scale of clusters and further reduce the number of iteration by using density-seeds and Omega-Cluster mechanism. The basic idea of OSP-Cluster is as follows.

The \( \text{Eps} \)-neighborhood of a spatial point, denoted by \( \text{NB}(p_i) \), is defined as the points in the neighborhood within radius \( \text{Eps} \). By considering the temporal variate, we extend this definition to the \( (\text{Eps}, \tau) \)-neighborhood.

Definition 7. \((\text{Eps}, \tau)\)-neighborhood. A \((\text{Eps}, \tau)\)-neighborhood \( (\text{NB}_\text{et}(p_i) \) for short) for spatio-temporal point \( p_i \) is defined as:

\[ \text{NB}_\text{et}(p_i) = \{ p_j \in P | \text{dist}(p_i, p_j) \leq \text{Eps} \land \text{time}(p_i, p_j) \leq \tau \} \quad (10) \]

where \( P \) is the entire set of spatio-temporal points, the \( \text{dist} \) function measures the spatial distance between tow spatio-temporal points, and the \( \text{time} \) function measures the time interval between tow spatio-temporal points.

Definition 8. \((\Omega)-\text{Combinable}\). Given the \((\text{Eps}, \tau)\)-neighborhood \( \text{NB}_\text{et}(p) \) of a spatio-temporal point \( p \), \( \text{NB}_\text{et}(p) \) and \( \text{NB}_\text{et}(q) \) are \( \Omega \)-combinable, denoted by \( \Omega(\text{NB}_\text{et}(p), \text{NB}_\text{et}(q)) \), if there are at least \( \Omega \) candidates such that both \( \text{NB}_\text{et}(p) \) and \( \text{NB}_\text{et}(q) \) contain them.

Definition 9. \((\Omega)-\text{Combinable spatio-temporal Cluster-OSP-cluster}\). An OSP-Cluster is the maximal set of \( \Omega \)-combinable spatio-temporal points. More specifically, given a subset \( P' \subseteq P \), \( P' \) is a \( \Omega \)-Cluster iff it satisfies \( \forall p \in P' : \exists o, p \in \text{NB}_\text{et}(o) \) and \( \forall \text{NB}_\text{et}(q), \text{NB}_\text{et}(q) \in P', \Omega(\text{NB}_\text{et}(p), \text{NB}_\text{et}(q)) \).

Thus, we apply the OSP-Cluster spatio-temporal clustering approach on spatio-temporal dataset grouping by time slots. This approach starts from conducting range query for each density-seed to detect the \( (\text{Eps}, \tau) \)-neighborhood, using \( \tau \), Eps and MinPts, to decide whether it is an OSP-cluster. Then it generates the initial OSP-cluster set with each spatio-temporal point inside marked as clustered and visited. Next, our method iteratively discovers new OSP-clusters in all unvisited points. If the number of the \( \text{NB}_\text{et} \) is less than \( \text{MinPts} \), the location is labelled noise; otherwise, locations are created as a new cluster if no neighbor is in an existing cluster, or it is merged.
with an existing cluster if more than $\Omega$ neighbors are in the same cluster. Further, we merge the clusters if the distance between them is close enough. Here, the distance between two clusters is measured by the centroid method, i.e., the average of latitudes and longitudes of tweets in the cluster. Because the distance threshold value $d_{mer}$ controls the size of a cluster, it should be relatively small to fit a cluster to a small-scale local event. Since the error of a geotagged tweet can be several hundred meters, the $d_{mer}$ is set as 200 meters to tolerate margins of the error.

Given $n$ as the number of all spatio-temporal points, $m$ is the number of non-clustered points after generating the initial cluster set and $v$ is the number of clusters detected from these non-clustered points, we have $m \ll n$ as the DSS set cover most of the potential locations where local events happen, and $v \ll n$. Thus, the complexity of this clustering algorithm is at most $m^2 + v \times (v + |DSS|) + |DSS|$, which is much smaller than the $O(n^2)$ of DBSCAN (or similar spatio-temporal clustering algorithms).

2) Local Event Detection and Important Keyword Extraction: For simplicity, the area of each OSP-cluster that used to indicate a potential local event is represented as its excircle and we can use the earliest time of post and the latest time of post as the duration of OSP-cluster and corresponding local event.

Local event has three components: the burst of spatial locality, burst of temporal locality and burst of terms. We can detect the first two components using the OSP-clusters, then we illustrate how to detect local event using the burst of terms in the OSP-clusters.

The local event can be described by a list of important terms, i.e., keywords, derived from the contents of geo-tagged tweets inside. The keywords of local events should not be terms that are generally mentioned at widely distributed locations (e.g., generally mentioned terms such as coffee or dinner) nor terms that are generally mentioned at any time (e.g. popular stations or sightseeing spots). Thus, we can use the burst of terms to detect a local event and use the keywords (i.e., burst terms) to represent the local event. Namely, we define a local event as keywords that occur significantly higher in a certain OSP-cluster (spatial and temporal localities) than in others. To detect the burst of terms, the non-burst terms should be filtered. Hence, we introduce a spatial keyword extraction and a temporal keyword extraction to filter out non-burst terms. One example is demonstrated in Fig. 3. Fig. 3a shows three OSP-clusters (A, B, C) discovered in a certain time slot. Given four terms ("Fire", "MCG", "Tennis", "Coffee") mentioned by the tweets in these OSP-clusters, each of them may be mentioned several times (i.e., co-occurrence) in one of the OSP-cluster. To filter the spatial non-burst terms, we search the posts at the same OSP-cluster area in the past times, as shown in Fig. 3b. Then we remove the frequently mentioned terms at the certain OSP-cluster area regardless of the time, such as "MCG" (a landmark in Melbourne city). To filter the temporal non-burst terms, we search the posts mentioned during the same time slot but distributed at different places, such as "Coffee". The terms like "Fire" and "Tennis" that occur neither at the same place regardless of the time nor at many places regardless of the time are selected as important keywords mentioned along with local events.

![Fig. 3. Example of local event detection and important keyword co-occurrences.](image)

Our approach aggregates the geotags of the training set of geotagged tweets according to the OSP-cluster that they are posted from, and the time interval they belong to. In the following sections, we describe the local event detection and keyword extraction approach.

1) Detecting temporal burst. To remove terms mentioned frequently (e.g., daily) in a certain area, we use the degree of temporal variance of term occurrences in each area, i.e., $IDF_i$, to represent the temporal burst. It searches the past posts in the OSP-cluster area during the past time slot and then counts the frequency of posts mentioned a given term in the same OSP-cluster area. We use the temporal IDF of a given term $w_i$ in an OSP-cluster at different time slot to measure the temporal burst, denoted by $IDF_i$.

$$IDF_i(w_i) = \log \frac{C_{TS'}(w_i)}{C_{TS'}(w_i)}$$

(11)

Here, $C_{TS'}(w_i)$ is the number of time slots mentioned $w_i$ in the given OSP-cluster area during a set of past time slots $T'$, e.g., the same time slot during the last week. The $IDF_i$ value will be small if a term is almost mentioned every day in the given area. Otherwise, this value will be higher if the term is a temporal burst term, which can be used as an important keyword.

2) Detecting spatial burst. Similarly, to remove terms mentioned several times at different area during the same time slot, we use the degree of spatial variance of term occurrences in each time slot to represent the spatial burst. It counts the frequency of tweets that contain a term in a certain time slot and then calculates spatial $IDF_s$ for a term $w_i$ in the time slot as below to measure the spatial burst.

$$IDF_s(w_i) = \log \frac{C_{OSP'}(w_i)}{C_{OSP'}(w_i)}$$

(12)

Here, $C_{OSP'}(w_i)$ is the number of spatio-temporal clusters in a certain time slot, and $C_{OSP'}(w_i)$ is the number of clusters mentioned term $w_i$ during the same time slot. The $IDF_s$ value will be small if a term is almost mentioned everywhere in the given area during a given time slot. Otherwise, this value will be higher if the term is spatial burst, which can be used as an important keyword.
After the tweet content is tokenized and analyzed using a tokenizer and POS Tagger tailored especially for Twitter, we only extract noun phrases in the document of the content of tweets in an OSP-cluster during a given time slot using morphological analysis. Note that successive nouns are treated as a phrase. Next, we identify co-occurrences of noun phrases and calculate $IDF_t$ and $IDF_s$ to determine whether the noun phrase is an important keyword or not. If both IDF values are more than the thresholds, the noun phrase is regarded as an important keyword. Note that, the thresholds are chosen at the IDF values that identify a certain proportion of terms as important keywords, e.g., the IDF values that allow 70% terms to be important keywords. Finally, we filter the OSP-clusters without important keywords returned and define the rest as the local events, described by the important keyword list.

(3) Weighting important keywords. Then, we calculate the co-occurrences of the keywords in each OSP-cluster: a dictionary containing the number of co-occurrence of each keyword in an OSP-cluster area during a certain time interval. Next, we generate a vector of important keywords (VIK) for each OSP-cluster during the certain time interval, along with the corresponding weights (WIK). The weight is used to evaluate the significance of each important keyword, and we use the TF-IDF approach for weighting keywords. Given georeferenced document $doc_{osp}$, consisting of a set of contents of geotagged tweets in an OSP-cluster $osp$, the TF-IDF value for the important keyword $w_i \in VIK_{osp}$ is calculated as follows:

$$TF - IDF(w_i, doc_{osp}) = \frac{C_{w_i}}{\sum_{w_j \in doc_{osp}} C_{w_j}} \times IDF_{w_i},$$

$$IDF_{w_i} = \log \left(\frac{|doc_{osp}(w_i)|}{|doc_{osp}|}\right).$$

Here, $C_{w_j}$ is the occurrence number of $w_j$ in $doc_{osp}$, $|doc_{osp}|$ is the total number of tweets in $doc_{osp}$ and $|doc_{osp}(w_i)|$ is the number of tweets mentioned $w_i$ in $doc_{osp}$. Then, the TF-IDF value can be used as the weight for the significance of each keyword.

Note that, we can use a variable proportion of all the important keywords, e.g., the top $k\%$ percentage of important keywords for each OSP-cluster according to the weight, to describe non-geotagged tweet or local event. To generate the keyword vector for the non-geotagged tweet $ng$, we apply the same process as before.

C. Event Matching-based Location Inference and Perturbation-C3

1) Similarity Evaluation: In this section, we aim to capture the two components of probabilistic model $P(e, ng, u)$ based on the similarity evaluation, and then use the $P(e, ng, u)$ to decide the best local event $e$ and used the OSP-cluster of $e$ as the geolocation of a given non-geotagged tweet $ng$. Specifically, we discuss the spatial similarity component $S1$ and the semantic similarity component $S2$ in the following sections.

Spatial Similarity Component ($S1$). In this part we capture $P(e, u)$, considering the geographical similarity (spatial proximity) of a user $u$ and a local event $e$. The geographical distance between historical visited geolocations by user $u$ and the geolocation of a local event $e$ affects the geographical similarity of $u$ and $e$, thus we use the distance between $l(u)$ and $l(e)$, denoted by $dis(e, u)$, to capture such geographical similarity.

We assume that each local event $e$ has a popularity $\varphi(e)$ and the spatial proximity of a user $u$ to a local event $e$ is captured by $\upsilon(e, u)$. $\varphi(e)$ is measured by the location entropy $e^H(e)$ (normalized H(e)) of the OSP-cluster of the local event based on visit distribution among users, and the $\upsilon(e, u)$ is measured by $e^{-k \times dis(e, u)}$, where $k$ is parameter that controls how fast the $\upsilon$ drops with distance. Note that the $dis(e, u)$ is captured by the Equation (14), where $GT_{u, t}$ is the historical geo-tagged tweets posted by $u$, e.g., the geo-tagged tweets from the last week, and the $dis(gt(l(e)))$ measures the geographic distance between a geotag $gt$ and the geolocation of local event $e$ during the same time slot $t_s$.

$$dis(e, u) = \{ \min_{gt \in GT_{u, t}}(dis(gt(l(e))))$, if $\exists GT_{u, t}$

The probability of generating a pair $(e, u)$ is directly proportional to the spatial proximity and the popularity of the local event and monotonically increase with the popularity and decreases with the spatial proximity between them.

$$P(e, u) \propto s1(e, u) = \rho(e) + \upsilon(u, e) = e^H(e) + e^{-k \times dis(e, u)}$$

After normalizing the $S1(e, u)$ by summing over all pairs of local events and users, the $P(e, u)$ is generated by:

$$S1 = P(e, u) = \frac{S1(e, u)}{\sum_{(e', u') \in E \times U} S1(e', w')}$$

Semantic Similarity Component ($S2$). In this work we do not aim to model the differences in the language usage of various users, i.e., we assume that the language model is independent of the user. Given the local event, the probability of observing a given text is independent of the user, i.e., $P(ng|e, u) = P(ng|e)$, thus $P(e, ng, u)$ is reduced to $P(e, ng, u) = P(e, u)P(ng|e, u) = P(e, u)P(ng|e)$.

The $P(ng|e)$ captures the probability of observing the words in the textual content of the non-geotagged tweet $ng$, given that the tweet is about local event $e$. In this section, we will calculate the $P(ng|e)$ based on the semantic similarity between the keyword vector of $ng$, denoted by $VIK_{ng, ts}$, and the keyword vector of a candidate OSP-cluster $osp$, detected during the same time slot with $ng$, denoted by $VIK_{osp, ts}$. Then, we extract the geolocation of a matched OSP-cluster as the geolocation resource for the geolocation of $ng$.

We calculate the magnitude of the $ng$ vector $VIK_{ng, ts}$ (associated with a weight vector $VIK_{ng, ts}$) by

$$magn_{ng, ts} = \sqrt{\sum_{i \in VIK_{ng, ts}} WIK_{ng, ts}[i]^2}$$
and the magnitude of each OSP-cluster $osp_i$ vector $VIK_{i,ts}$ (associated with a weight vector $WIK_{i,ts}$) that represents a local event by

$$magso_{i,ts} = \sqrt{\sum_{i \in VIK_{i,ts}} WIK_{i,ts}^2} \tag{18}$$

The similarity between the $VIK_{ng,ts}$ and $VIK_{i,ts}$ are calculated over all the keywords that appear in both the vectors, obtained by the Equation (19), using the weight of each vector, i.e., $WIK_{ng,ts}$ and $WIK_{i,ts}$.

$$Sim(ng, i) = \frac{\sum_{j \in VIK_{i,ts} \cap VIK_{ng,ts}} WIK_{ng,ts}[j] \times WIK_{i,ts}[j]}{magng_{ng,ts} \times magi_{ts}} \tag{19}$$

Then a similarity score list is used to store the similarity value of each pair of non-geotagged tweet and OSP-cluster, which will be normalized over the sum of all similarities to represent the probability $P(ng | e)$, where the local event $e$ is represented by its OSP-cluster $osp_i$ (i for short).

$$S2 = P(ng | e) = \frac{Sim(ng, i)}{\sum_j Sim(ng, j)} \tag{20}$$

The probability of a non-geotagged tweet $ng$ posted in OSP-Cluster $osp_i$ (to represent the local event $e$) can be captured by $P(e, ng, u) = S1(i, ng) \times S2(i, ng)$. Finally, we choose the geolocation(s) of OSP-cluster(s) with the highest probability(ies) as the geolocation of $ng$. The similarity calculation approach is presented in Algorithm 2.

**Algorithm 2: Similarity Evaluation and Probability Inference**

**Input:** A set of geotagged tweets $GT$, a set of non-geotagged tweet $NG$.

**Output:** The estimated geolocation.

for $ts$ in $TS$ do  

1. $NG_{ts} \leftarrow \{ng \in |GT| \text{ in a timeslot } ts\}$
2. $GT_{ts} \leftarrow \{gt \in |GT| \text{ in } ts\}$
3. $LEC_{ts} \leftarrow \text{run C2 on } GT_{ts}$
4. for $i$ to $|NG_{ts}|$ do
5.  
6.  
7.  
8.  
9.  
10.  

For simplicity, we perform the geolocation inference procedure in the term of the time slot. Namely, we estimate the geolocation for the tweets at a certain time slot at one time to avoid repetitive spatio-temporal clustering and event matching procedures. In addition, we do not try to estimate the geolocation for all non-geotagged tweet, we only focus on the non-geotagged tweets that can be discovered a high probability geolocation in a specific area (e.g., Melbourne City). Note that, we determine the city with the highest probability for the non-geotagged tweet $ng$ at first, and then determine the OSP-clusters within that city. For example, we can start with the city granularity using the spatial proximity $\upsilon(city_i, u)$ defined in section 3.3.1 or the home location of the user, and then scale to the OSP-cluster granularity.

2) Private Possible OSP-Cluster Selection: In this step, the privacy-preserving mechanism is embedded in the possible OSP-cluster selection.

It consists of two steps: possible set construction and private perturbation.

First, we sort the possible OSP-clusters according to the probability and select some most possible candidates as the possible set for $ng$. Specifically, set of sorted candidates with the cumulative probability is greater than $1 - \alpha$ are chosen to construct the $\alpha$-dataset. If there are less than $k$ possible OSP-clusters for $ng$, the geocoding procedure will be skipped.

Then, the private perturbation is used to select a possible candidate from the $\alpha$-dataset. Specifically, we randomly choose a possible candidate as the geolocation source of $ng$ from its $\alpha$-dataset using exponential mechanisms [13] to satisfy differential privacy. The chosen probability for each candidate of the possible set is calculated based on the sensitivity and the score function. The probability value $P(e, ng, u)$ derived from similarity value between the keyword vector and that of an local event that is represented by its OSP-cluster is used by the score function $score$ for all candidate in the possible set, which is defined as:

$$score(i, ng) = P(e, ng, u), i \in \alpha - dataset \tag{21}$$

The maximum change in the probability that any $i \in \alpha - dataset$ can be selected as the geolocation of $ng$ is used as the global sensitivity (GS). For simplicity, GS can set as 1. Based on the $q$ and GS, the probability for each $i \in \alpha - dataset$ is calculated as follows:

$$Pr(osp_i) = \frac{\exp(\epsilon \times score(i, ng))}{\sum_j \exp(\epsilon \times score(i, ng))} \tag{22}$$

Here, $\epsilon$ is the privacy budget used in the private possible candidate selection. Based on the chosen probability, an OSP-cluster is randomly chosen to present the geolocation of $ng$.

D. Privacy Analysis

To prove P-GENT satisfies $\epsilon$-differential privacy. We use an exponential mechanism to perturb the OSP-cluster selection with given privacy budget $\epsilon$ and global sensitivity GS, as the output perturbation. The private selection function, denoted by $M$, for location inference of each non-geotagged tweets satisfies $\epsilon$-differential privacy is proved as follows.

As the score function $score(D_1, r) - score(D_2, r) \leq GS$ for any $D_1, D_2$ differing at most by one
Here, IS is the proportion of the set of re-identified results. The distance between them is less than a threshold), i.e., the number of real estimated geolocation re-identified by the perturbed data users. The recall is defined as:

\[
\text{Recall} = \frac{|\text{TGT}|}{|\text{EGT}|} \text{, } \text{Precision} = \frac{|\text{TGT}|}{|\text{IST}|} \text{, } F1 = \frac{2 \times P \times R}{P + R}
\]

Here, TG is the number of correctly geocoded tweets of the test dataset, i.e., the distortion between the estimated location and the original location of geotag is within 500 meters, EGT is the number of testing tweets that we can estimate a geolocation, i.e., the distortion between the estimated location and the original location of geotag is within 1000 meters, and TGT is the total number of tweets in test dataset. Note that we only associate a possible geolocation for tweet has candidates in its α-dataset within a threshold, e.g., 1000 meters.

3) Competitors: We experimentally evaluate the private geocoding approach into three aspects, i.e., the privacy and utility analysis of the estimated geolocation after perturbation when varying privacy budget ε, the accuracy analysis of the estimated geolocation, and the performance of the OSP-Cluster spatio-temporal clustering.

In order to illustrate privacy and utility trade-offs of the perturbed estimated results over varying ε, a range of experiments were conducted. Specifically, we use recall and ADE to illustrate privacy and utility analysis.

For accuracy analysis, we implement four competitors, namely, the grid-based inference (“grid” for short) as baseline to evaluate the efficiency of our seed-based approach, the inference approach use the most possible OSP-cluster to be the estimated geolocation (“most” for short) as baseline to evaluate the efficiency of the threshold-based mechanism(“Th” or “±ε%” for short), and the inference approach with and without perturbation mechanism to evaluate the utility retained after perturbation and compared to other competitors (“With” and “Without” for short respectively).

For clustering performance analysis, we use the runtime and the number of detected spatio-temporal clusters as the evaluation metrics for effectiveness and efficiency of the OSP-Cluster spatio-temporal clustering algorithm compared to ordinary one such as extended DBSCAN for spatio-temporal clustering (DBSCAN) varying clustering parameters. We also compare the OSP-Cluster clustering with and without density-seed mechanism (Seed and Omega-2 respectively) both with Ω = 2 as well as the OSP-Cluster clustering approaches with different Ω values.

C. Evaluation Analysis

1) Privacy and Utility Analysis: The initial step is to find the threshold that can be used to declare whether an estimated
location is re-identified or not after using the private OSP-Cluster selection approach. An optimal threshold can be used to ensure a high recall and associated low distance between the estimated geolocation using the ordinary and the private OSP-Cluster selection approach.

Here we set the minimum distance between them at which the recall is higher than 70% as the threshold, as shown in Fig. 4a. We evaluate the threshold of the private mechanism when varying $\epsilon$ and $\alpha$ in Fig. 4b and 4c. Using the method described above, the thresholds can also be determined and used as the indicator for average distortion of perturbation. It is clear that the threshold increases as $\epsilon$ increases. Namely, privacy and utility are trade-offs, and a higher degree of privacy preservation causes lower distortion level.

![Fig. 4. Effect on recall and threshold.](image)

In addition, the threshold results in an increasing trend as $\alpha$ declines, because a smaller $\alpha$ will boost the number of potential candidates that with larger distortion and then the threshold will be larger as well. This shows that the utility of perturbed data will be improved by decreasing the degree of privacy preservation and increasing $\alpha$ using the $\alpha$-dataset mechanism.

![Fig. 5. Effect on ADE.](image)

Fig.5 shows the ADE with varying $\epsilon$. As $\epsilon$ increases, i.e. the privacy level weakens, ADE decreases obviously since less noise is added. As $\alpha$ declines, namely $1 - \alpha$ increases, and more candidates exist in the ($\alpha$)-dataset, ADE increases. This confirms the trade-offs mentioned above.

2) Accuracy Analysis: In Fig.6, we present the average accuracy (generally represented by precision) that the P-GENT achieves among all timeslots of the testing dataset, varying the proportion of the important keywords adopted while only considering the most possible candidate of $\alpha$-dataset for estimation. As shown in Fig.6, we find that the best average accuracy is 47.61% when choosing 80% of the important keywords to generate the $\textit{VIK}$ and using the P-GENT without perturbation mechanism (denoted by $\textit{Without}$), followed by the accuracy of the P-GENT with perturbation mechanism (denoted by $\textit{With}$). For the $\textit{With}$, the best accuracy is almost 43% when using 100% of the important keywords, which shows an increasing trend as the proportion increases. After evaluating the P-GENT approach with and without perturbation, we compared them to the grid-based approach baselines, using the ($1^\circ, 1^\circ$) grid granularity. As illustrated in Fig.6, we can see that the accuracy of $\textit{Without}$ is over 40% better, and the accuracy of $\textit{Without}$ is over 30% better. The improvement is due to the fine-grain granularity we use, i.e., the granularity is shrunk from the large grid to the finer OSP-cluster. Furthermore, probably due to the finer granularity, the results after perturbation are still efficient. In addition, a randomly choosing approach is also used for comparison. we randomly choose a grid that has geotags at the training datasets to be the estimating result. The average accuracy of such algorithm is under 3%.

![Fig. 6. Accuracy on all time slots.](image)

In order to measure how the proportion of the keywords used effects the execution times, we measure the average execution time required for per timeslot for estimation varying the proportion of keywords used. We find that the execution time increases as the proportion goes up. Further, less time used by both $\textit{Without}$ and $\textit{With}$ than grid. This is due to the fact that the P-GENT not only reduces the number of geographic reference candidate but also further eliminate many of them by filtering mechanism. Moreover, as the accuracy does not always increase as more proportion of keywords are used, thus an appropriate proportion should be decided experimentally.

![Fig. 7. Accuracy using dynamic matching similarity thresholds.](image)

Next, we evaluate whether a probability threshold mechanism used in event matching approach can improve the inference accuracy or not, compared to choose the candidate with the largest probability as the geolocation reference for each non-geotagged tweet. Thus, we evaluated our methods when using a dynamically defined probability (event matching similarity) threshold. The threshold we have chosen to use is automatically calculated by the results of the 4-hour timeslots of the previous days. As a result, we have 6 user-free dynamic
thresholds that are calculated by taking the average similarities of the previous respective timeslots. By introducing the threshold mechanism, the recall is reduced but the precision increases. In Fig. 7, we evaluate the precisions, recalls and F1 values with and without of the threshold mechanism in P-GENT without perturbation. Experiments are conducted by using no threshold (most), the exact dynamic threshold (Th), and the exact threshold value + − 10% and + − 20%.

As we can see from Fig.7, the recall is reduced from 37% to 21% after introducing threshold mechanism and it will further decrease as the threshold increase. The reason is that some non-geotagged tweets become non-estimated as all candidates of its α-dataset have less probability than the threshold. However, the precision is improved from 36% to 73% after introducing threshold mechanism and it will further decrease as the threshold increase, up to 92.4%. As we can see from the F1 measures, although the precision is measured by precision, it does not mean that the higher the threshold, the more efficiency of estimation. Thus, an appropriate threshold should be decided experimentally.

3) Clustering Result Analysis: In this section, the runtime and number of detected spatio-temporal clusters are adopted as an evaluation metric for effectiveness and efficiency of the OSP-Cluster clustering algorithm compared to DBSCAN and other competitors.

![Fig. 8. Clustering result analysis.](image)

Fig. 8a illustrates the performance of the OSP-Cluster clustering algorithm when varying the sizes of the datasets compared to other approaches. It shows that the performance of the OSP-Cluster clustering approach using density-seeds (Seed) is the best, followed by the OSP-Cluster clustering approach without using density-seeds (Omega-2). We can see that Seed significantly reduces execution time, which means the density-seed mechanism can significantly speed up the clustering. By comparing the Omega-Cluster-based clustering with DBSCAN, the former can reduce more execution time, both using Ω = 1 and Ω = 2. Moreover, we find that the larger the datasets, the better the performance achieved by OSP-Cluster clustering algorithm due to its density-seed and the Ω-Combinable relation. We see that the performance of the OSP-Cluster clustering is less sensitive to the dataset size than other competitors. These comparisons can confirm the time complexity was reduced via applying the anytime scheme.

Next, the clusters number was adopted to address the parameters insensitivity property of the OSP-Cluster clustering algorithm and find the appropriate parameters to achieve better clustering results. Fig.8b and 8c indicate the average number of clusters discovered by OSP-Cluster clustering approach with varying Eps and MinPs, compared to DBSCAN. We can see that the number of discovered clusters declines as Eps and MinPs increases in both curves, which demonstrates that a proper Eps and MinPs should be selected to ensure the utility. Based on these experiments results, the proper parameters can be decided. Moreover, the number of clusters detected by OSPmega-Cluster is significantly more than DBSCAN at different parameter settings. Further, the performance of OSP-Cluster is more stable than DBSCAN with different parameters values. This is due to the efficiency of the notion of a Ω-combinable relation used in OSP-Cluster compared to the density-connected relation used in DBSCAN.

In addition, Fig.8d shows the number of clusters by varying Omega. As Omega increases from 1 to 3, the number of discovered clusters has an increasing trend. When Omega is set smaller, bigger clusters may be generated due to the loose merging constraints and the number of clusters was reduced.

These comparisons can confirm the time complexity is reduced and the performance is improved by applying density-seed mechanism and using the Ω-combinable relation.

V. RELATED WORK

Several existing works have studied on estimation geo-location for non-geotagged tweets. [4], [5] applied a probabilistic model that used the probabilistic distribution of users local words for location inference. Mean Field Variational inference and Kullback-Leiber divergence were used to develop geographic and topic models for location estimation in [14]. Lists of words that represent a location by using Latent Dirichlet Allocation (LDA) were used to location estimation in [14]. [16] used a topic model in which the distribution of topics is conditioned on geographical location. [17] combined text and geographic information, in which each topic generates geospatial coordinates from two topic-specific Gaussian distributions. However, most of these estimation approaches are based on analyzing the text in the tweets, which operate at a huge time and space scale.

Our work differs from these in the following ways: we use a spatio-temporal burst of geo-tags to detect local events and use the similarity between the contents of non-geotagged tweets and documents of local events to estimate the location. Several existing works have studied the local events by analyzing the tweets. [18] discovered specific kinds of local events (earthquakes and typhoons) and estimated the geolocation where they were occurring. Predefined keywords (e.g. earthquake) were used to filter non-relevant tweets and then the context of the keywords is used to estimate the probability that the local event is actually occurring. [19] detected the epidemic of the flu using a keyword-matching method to analyze the tweets referring to the flu. However, these works can only solve limited kinds of local events using a few predefined keywords.
In our work, we focus on any types of local events. Moreover, we provide much finer inferring granularities than what the existing approaches can afford, as well as we incorporate the spatial proximity between a user and a local event, popularity of the local event and the weight of keyword for object matching.

Further, estimating the fine-grained geolocation for non-geotagged tweets raises many privacy issues, as well using the spatio-temporal data. Such issues are discussed in [20]. On the other hand, differential privacy presented in [8] has been proved to be roused. To the best of our knowledge, there is no research on combining density-seed region identifying, spatio-temporal clustering, non-geotagged tweets geocoding, and privacy-preserving mechanism under differential privacy into a single task.

VI. CONCLUSIONS

We propose a framework P-GENT to effectively estimate the geographic location of non-geotagged tweets with privacy protected and then release perturbed results for public use. The approach improves the efficiency and the accuracy of the estimated geolocations from three aspects: (1) The sparseness and time complexity issues are alleviated by using density-seeds discovered by an automatic cluster center detection approach and using the Omega-Cluster-based clustering algorithm; (2) A local event matching model is used to achieve a fine-grained location estimation of non-geotagged tweets, based on three similarity quantities consisting of the content similarity weighted by TF-IDF, spatial proximity between user and event’s locations as well as the popularity of the locations of local events. (3) An OSP-Cluster spatio-temporal clustering and keyword extraction approach is proposed to detect local events from geo-tagged tweets and achieve a finer estimating granularity. P-GENT finally implements a private possible OSP-cluster selection approach to protect the privacy of estimated results before releasing for the public use. We conduct the experimental evaluations with real-world data to confirm the utility and accuracy of our approach, remained after using privacy-preserving mechanisms. We also conduct comparisons to confirm the time complexity can be reduced and the performance can be improved by P-GENT. We believe P-GENT is an important step towards designing a practical differentially private geocoding approach.

REFERENCES


Minerva Access is the Institutional Repository of The University of Melbourne

Author/s:
Shuo, Wang

Title:
Exploration and protection of location privacy in online social networks

Date:
2018

Persistent Link:
http://hdl.handle.net/11343/212415

File Description:
Appendices-author accepted manuscripts-Chapter 10

Terms and Conditions:
Terms and Conditions: Copyright in works deposited in Minerva Access is retained by the copyright owner. The work may not be altered without permission from the copyright owner. Readers may only download, print and save electronic copies of whole works for their own personal non-commercial use. Any use that exceeds these limits requires permission from the copyright owner. Attribution is essential when quoting or paraphrasing from these works.