Soc. Netw. Anal. Min. (2016) 6:109
DOI 10.1007/s13278-016-0417-y

ORIGINAL ARTICLE

Supporting geospatial privacy-preserving data mining of social media

Shuo Wang[1] · **Richard O. Sinnott**[1]

Received: 29 September 2015 / Revised: 1 November 2016 / Accepted: 14 November 2016 / Published online: 26 November 2016

Abstract With the global adoption of smart mobile devices equipped with localization capabilities and broad popularity of microblogging facilities like Twitter, the need for personal privacy has never been greater. This is especially so with computational and data processing infrastructures such as clouds that support big data analysis. Differential privacy of geospatially tagged data such as tweets can potentially ensure that degrees of location privacy can be preserved while allowing the information (tweet contents) to be used for research and analysis, e.g., sentiment analysis. In this paper, we evaluate differential location pattern-mining approaches considering both privacy and precision of geo-located tweets clustered according to Geo-Locations of Interest (GLI). We consider both the privacy protection strength and the accuracy of results, measuring the Euclidean distance between centroids of real GLIs and obfuscated ones, i.e., those incorporating privacy-preserving noise. We record the performance and sensitivity of the approach. We show how privacy and location precision are trade-offs, i.e., the higher the degree of privacy protection, the fewer the GLIs will be identified. We also quantify these trade-offs and their associated sensitivity levels. We illustrate the work through a big data case study on use of Twitter data for traffic-related data protection.

✉ Shuo Wang
shuow4@student.unimelb.edu.au

Richard O. Sinnott
rsinnott@unimelb.edu.au

[1] Department of Computing and Information Systems, University of Melbourne, Melbourne, Australia

Keywords Differential privacy · Location privacy · Tweets · Privacy spatial decomposition · Pattern mining

1 Introduction

Social networks with location awareness such as Twitter are geared toward allowing users to share general information through 140 character strings—so-called tweets. Twitter has become a global phenomenon with over 400 million tweets made daily. Many users are unaware that often the geo-location of the tweet is also recorded, i.e., where they actually tweeted from and at what time they tweeted. This has obvious privacy issues. As an illustration of this, Fig. 1 illustrates how Twitter data can provide more information on individuals than they would ever have thought possible: tracking them throughout the day to potentially discover many aspects of their lives; where they live; where they travel; what they are doing; what time of day they are actually doing it, etc. Furthermore, once an individual has tweeted they can in principle be tracked directly and potentially forever using the Twitter Search API that is itself openly (programmatically) accessible.

It could be argued that location-based information itself should be removed at source, e.g., by Twitter; however, there is an increasing demand to localize the aggregated analysis of Twitter data. This can be for real-time information on a variety of issues: congested transport routes around cities; using Twitter data as the basis for early warning health outbreaks (avian flu, Ebola virus outbreaks); natural hazards (bushfires, earthquakes, floods) among many other scenarios. Given this and the potential for the many positive uses of such data, supporting degrees of privacy in aggregated geospatial settings, is highly

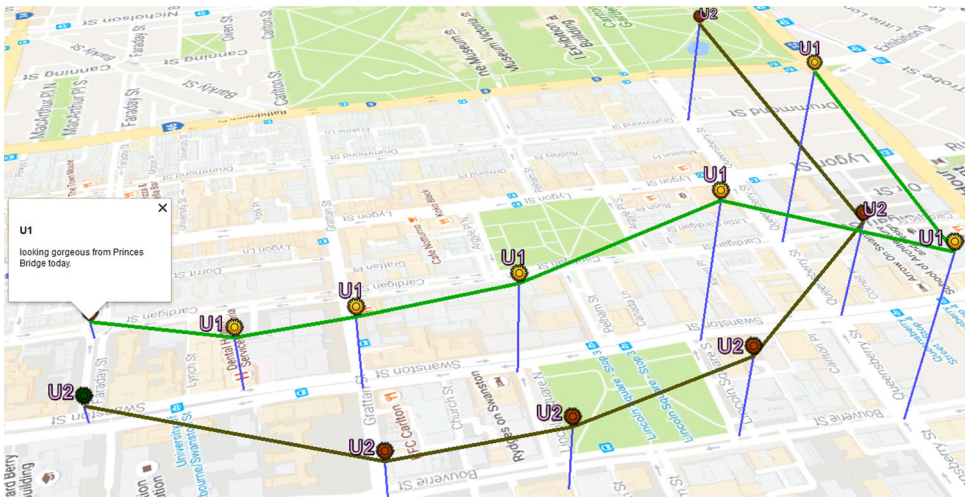


Fig. 1 Geospatially tracking a small sample of Tweeters around Melbourne (*color codes* are individual Tweeters, and *vertical lines* represent increasing times of day)

desirable. Location-based services are increasingly popular and not restricted solely to Twitter (Hasan et al. 2013). Many mobile applications capture location-based information and often are deliberately designed to use this information. Indeed, Twitter provides a "Nearby" application for users to find friends/followers who are tweeting in a particular nearby locality. However, there is a clear need for more control over the privacy of shared information—especially the potentially unforeseen privacy consequences such as user location tracking as shown in Fig. 1. Thus, users might be happy to acknowledge that the tweets they make are for public consumption and hence non-private by their nature; however, where and when they make them and the consequences that can arise through this have given rise to increasing demands for privacy (Sadeh et al. 2009). The availability of major computational resources such as clouds and technologies such as NoSQL data resources and big data processing algorithms such as MapReduce and ElasticSearch now allows mining and analysis of such data at an unprecedented scale. Given this, it is meaningful to explore behavioral analysis and pattern mining of location data and ways to obfuscate this sensitive information, especially as it could be used for malicious purposes against Tweeters and potentially their followers (Yu et al. 2011).

Threats to reveal supposedly anonymous individual behaviors are exacerbated when attackers possess degrees of background knowledge. Consequently, several solutions have been proposed to mine location data with differential privacy (Arik and Schuster 2010). In recent years, differential privacy (Dwork 2006) has been widely used for the protection of location-based data. In these solutions, it was shown that location privacy could be preserved by adding moderate degrees of noise based on an appropriate degree of required location obfuscation, while supporting degrees of service for other location-based services. The advantage of differential privacy for location privacy is that it allows to protect individual location information while still allowing the data to be used for analysis and/or mining. Solutions that can limit the dangers of leaking location privacy would encourage more users to share their location information. Hence, a large amount of meaningful work with social utility could be carried out with improved aggregate geospatially coded Twitter data, e.g., pandemics and natural disasters often rely on social media and being able to undertake pattern mining to extract knowledge such as the Geo-Location of Interest (GLI) with "safe" degrees of privacy preservation.

While these solutions have demonstrated that classical differential privacy can be achieved, it is sometimes difficult in practice to introduce suitable degrees of noise. Too much noise and the aggregation of geospatial information render the data useless for location-based analysis; too little noise and the dangers of privacy violations exist. To tackle this, in this paper a differential privacy-based spatial partition is adopted and combined with a spatial clustering algorithm focused on mining locations of interest. Specifically, a geo-location database extracted from tweets from Twitter is established and populated with geospatial location where optimal quad-tree spatial decomposition is used with differential privacy to discover targeted locations of interest. Building on this, a recursive density-based clustering algorithm (Changqing et al. 2004) is used for clustering likely regions, i.e., privacy-protected ones, with actual locations of interests. To achieve this, a Laplace noise mechanism is introduced to obfuscate tweet locations into targeted privacy-preserving regions. Finally, we contrast noise-based privacy-preserving GLI clouds with actual (i.e., non-privacy protected) tweet-based GLIs to analyze the overall privacy and the accuracy of the solutions. The Euclidean distance between real GLIs (the actual/original tweet location) and noise-induced ones

together with the number of similar neighborhoods surrounding real GLIs and noise-induced ones is analyzed.

The main contributions of this paper are as follows: (1) development of an adaptive privacy-preserving special decomposition solution OptQ-SDDP supporting geometric privacy budget to improve utility, (2) supporting GLIs with differentially private guarantees using intelligent parameter settings and (3) ensuring private GLI pattern-mining solutions over large space–time domains comprising realistic location challenges facing large-scale social networks with a range of comprehensive evaluation metrics.

The rest of this paper is structured as follows. Section 2 provides an overview of related work with a focus on location privacy-preserving, and the advantages of differential privacy compared with other methods and works used for location privacy preservation. Section 3 describes the foundations for GLI pattern mining of tweets; the ideas and mechanisms that underpin differential privacy. Section 4 presents the evaluation metrics adopted in the work. Section 5 presents the experimental results of the privacy-preserved Twitter analysis focused on traffic events reported through social media. Finally, Section 6 draws conclusions on the work as a whole.

2 Related work

K-anonymity (Sweeney 2002) has been widely used to protect privacy in location-based systems based on the hypothesis that it is impossible for attackers to differentiate an individual, from k other different individuals. When it is used for location privacy preservation, the set of k points should be indistinguishable Chow et al. (2009). There are many ways to implement this method, such as introduction of dummy locations Kido et al. (2005) and cloaking Xue et al. (2009). The former solution adds $k-1$ properly selected dummy points and uses both the real and dummy locations for analysis. Cloaking uses artificial cloaking areas that include k points sharing some property of interest for analysis. The drawback of k-anonymity is that it is built on assumptions about the quantity of a potential attacker's auxiliary knowledge, i.e., the approach fails if dummy locations can be distinguished from real locations by attackers. Although some improvements have been proposed [i.e., l-diversity (Ashwin et al. 2008), t-closeness notion (Ninghui et al. 2007)] considers ubiquity, congestion and uniformity when dummy points are generated, e.g., to make them look more similar to real locations or taking an individual's auxiliary information into consideration to construct a cloaking region, and (Abul et al. 2008), Hu et al. (2010) and Terrovitis and Mamoulis (2008) put forward the (k, δ)-anonymity pattern, which depends on inaccurate sampling and location systems, where δ represents the possible positioning inaccuracy. It focuses on amending trajectories through space translation to make k different trajectories co-exist in a cylinder of radius δ. It reveals the problem of k-anonymization of a trajectory database relating to sensitive events. It aims to ensure that at least k users are able to get access to every event. In particular, this work proposes a new generalization mechanism known as local enlargement, which works better than traditional level or partition-based generalization. However, there are also some defects that can be attacked. For example, assumptions cannot be made regarding how much additional information an attacker might have. Differential privacy can avoid these defects, as it defines rigorous obfuscation (privacy preserving) models and has nothing to do with the attacker's potential auxiliary information about an individual.

Differential privacy was introduced by Dwork et al. (2006). It ensures that useful information can be inquired and mined from a statistical database comprised of individually identifying information, while protecting a given individual's privacy. It provides privacy guarantees as to whether or not a single element is present inside a database or not without explicitly identifying the individual. Several efforts have explored how to apply differential privacy to protect location privacy. One example is to support Geo-indistinguishability (Andrés et al. 2013) using a disturbance technique, whereby a Laplace distribution including stochastic noise is used to obtain Geo-indistinguishability. To evaluate the capacity of Geo-indistinguishability to defend a user's points of interest (POIs), Primault et al. (2014) collected real mobility traces from two diverse datasets and demonstrated that Geo-indistinguishability is often inadequate because attackers can distinguish at least 63% of users although the location is often vague. Jiang et al. (2013) have used differential privacy to protect trajectories of ships by generating and adding noise to trajectories. They explored three ways to add noise: adding global noise to the whole trace; adding noise to each point (x, y) independently and adding noise to each x- and y-coordinate independently. The available privacy-preserving data publishing methods coming from partition-based privacy models, like k-anonymity, may not protect privacy sufficiently; however, they identify that differential privacy approaches may well meet this objective. There are two main types of space splitting techniques used in partition-based privacy models: data dependent and data independent. The data-independent method does not consider the distribution of the points in space and achieves decomposition through recursively splitting the areas, e.g., quad-trees split the areas into four equal squares. Quad-tree-based solutions split regions based on point distributions. Several other techniques distinguish points in space; for example, Hilbert R-Trees seek out points in a given space

and splits the regions again using the Hilbert curve. Ho and Ruan (2011) introduced a classical approach to applying differential privacy to location data mining focused on protecting the privacy of the outcome of an aggregate function but not the entire dataset. To achieve this they used an approach based on equal privacy distribution, which leads to lower utility. In addition, there is no sufficient evaluation on utility and privacy of the solution. Xiao et al. (2010) put forward a novel method for spatial data partitioning. Xiao et al. (2011) developed a wavelet-conversion method suited for relational data to reduce noise magnitude, instead of adding independent Laplace noise. It questions each probable association of attribute values and establishes a generalized result according to the perturbed outcomes. The algorithms in Dewri (2012) were developed to deal with all kinds of entries in the area, causing extensibility of the trajectory data background.

3 Differential privacy-preserving GLI mining

3.1 Background

In this section, we contrast various methods that have been used to achieve privacy preservation, focusing especially on techniques used for geospatial data privacy. We introduce *k*-anonymity and differential privacy and discuss their advantages and disadvantages. We define and describe GLI and present recent approaches for mining with privacy protection to discover location information.

3.1.1 Differential privacy

Differential privacy was proposed by Dwork et al. (2006). It is based on the idea that valuable knowledge can be gained from datasets without disclosing sensitive information. It offers rigorous privacy assurances that one individual cannot be recognized whenever this individual is in or is deleted from the dataset, i.e., the results will not change sufficiently to identify the difference.

The formalized definition of differential privacy is that if an individual is deleted from a database, there is no output that becomes obviously changed. Specifically, a private function K with ε-differential privacy for databases D_1 and D_2, differing at most one element from each other, satisfies differential privacy if for all outcomes of the database S ($S \subset$ range (K)) there is:

$$Pr[K(D_1) \in S] \leq e^{\varepsilon} Pr[K(D_2) \in S] \tag{1}$$

A mining algorithm O provides ε-differential privacy if for any two datasets D_1 and D_2 that differ in a single entry and for any a in the database (Nissim et al. 2007) then:

$$\left| \log \frac{p(O(D_1) = a|D_1)}{p(O(D_2) = a|D_2)} \right| \leq \varepsilon \tag{2}$$

In (2), $O(D)$ is the output of the algorithm, p is the probability density and ε is a value that represents the privacy leakage. ε-Differential privacy can be achieved by the addition of random noise whose magnitude is chosen as a function derived from the largest change a single participant could have on the output to the query function. This often referred to as the sensitivity of the function (Nissim et al. 2007).

ε-Differential privacy can be realized by introduction of noise in several ways. One example (adopted here) is through introduction of a Laplace noise mechanism (Lap(σ)), whose magnitude is related to the variation that the removal of a single participant can cause on the output. The maximum query output variation when removing an element of the database is represented by the global sensitivity of a given query (Dwork et al. 2006).

$$\Delta f = \max_{D_1, D_2} \|f(D_1) - f(D_2)\|_1 \tag{3}$$

For a given function f: $D^n \rightarrow R^{\mathrm{d}}$ (where R^{d} is a d-dimensional vector) the global sensitivity is shown in (3). Differential privacy has two important properties (McSherry 2009):

- Sequential composition: The differential privacy provided by a sequence of mechanisms M_i on an input domain D is $\sum_i \varepsilon_i$.
- Parallel composition: If every mechanism M_i acts on a disjoint subset $D_i \subset D$, the privacy provided will be (max(ε_i))-differential privacy for all M_i.

Nissim et al. (2007) introduced local sensitivity to improve the limitations of global sensitivity, i.e., it cannot reflect a possible function's insensitivity to individual inputs due to an overload of noise, because it is concerned with a specific instance of the database. For f: $D^n \rightarrow R^{\mathrm{d}}$ (where R^{d} is a d-dimensional vector) the local sensitivity of f at x is:

$$\Delta f_{\mathrm{local}} = \max_{x,y:d(x,y)=1} \|f(x) - f(y)\|_1 \tag{4}$$

The value of this function is calculated over a specific x and all the possible neighbor databases y that differ from x by only one element (Ho and Ruan 2013).

3.1.2 Mining Geo-Locations of Interests in tweets

Because of the differences in the location of tweets (i.e., they will typically have different latitude/longitude given), any given tweet geo-location can be described as $\mathrm{loc}_i^{K_i} = \{S_1, S_1, \ldots, S_{K_i}\}$, where s is defined as one place (location) where the user tweets. Each $s_j = (lat_j, lon_j,$

$time_j$), where lat_j is the latitude, lon_j the longitude, $time_j$ the location in time that a user tweets.

- A Users Location of Interest (ULI) is defined as a geospatial and temporal circle with radius $\geq r$ where the user location dataset loci are contained within the circle of radius r. The higher the value of r, the greater the privacy is.
- A Geo-location of Interest (GLI) is an area containing at least m User Location of Interest (ULI) where each user has more than r' tweets location marked in that area.

One way to consider a ULI and a GLI in the context of Twitter is that a ULI is a cloud of uncertainty of where a tweet actually took place. This cloud covers both location and temporal dimensions. Some users might demand high privacy in which case the coverage (radius) of the cloud is increased. GLI can be considered as areas of high concentrations of privacy-preserved tweets. It is noted that the actual Tweet content, e.g., particular hashtags such as #Airport #University #Library, can also be used to filter and cluster tweets of interest. A GLI can be used to identify correlations between users and events or activities without explicitly identifying the location of the event.

3.2 Data harvesting and preprocessing

All of the data harvesting and preprocessing is implemented on the Australian National eResearch Collaboration Tools and Resources (NeCTAR) Research Cloud (www.nectar.org.au). The harvester itself implements a RESTful client that connects to the Twitter Search API. The returned tweets are processed and incorporated into the NoSQL database (CouchDB). This processing involves removal of tweets that do not explicitly have geospatial information included (latitude/longitude). CouchDB was selected in part as it natively supports MapReduce.

The system supports elastic scaling, and more harvesters can be deployed across cloud resources. Four medium-sized virtual machines with 8-Gb memory and eight virtual CPUs with 250-Gb volume storage and 100-Gb object storage were used as the basis for the work.

3.2.1 *The structure of cloud-based virtual machine instances*

The structure of the infrastructure used across the NeCTAR cloud contains six virtual machine instances to harvest data from Twitter, with two VMs for stream API harvesting, two for REST API harvesting, one for CouchDB and one for processing data. The tools and systems used to deliver the infrastructure included shell scripts, Ansible, OpenStack nova clients, public/private keys and OpenStack RC files. Specifically, these were used to automatically build, deploy and configure instances and volumes over the NeCTAR Cloud. OpenStack nova clients allow for instance creation and association of security and configuration information, e.g., to create security groups to connect instances with CouchDB to store harvested data. Five small instances were used for the harvesters and one medium instance used for CouchDB with one small virtual machine for the user interface (UI). Volumes were attached to instances through scripting languages utilizing Ansible and execution of yml files in the local host and subsequently across the cloud resources. A set of IP address of each virtual instance is returned.

3.2.2 *Cloud-based data harvesting*

The focus of data harvesting is to obtain tweets from the Twitter API according to the geo-location coordinates specified and saving the data to a centralized CouchDB instance with given IP address and the associated name of the database.

The Twitter Stream API and Search API were utilized concurrently. However, different harvesting programs can result in collecting duplicate tweets. To avoid duplications, the Tweet ID was used as the document ID in CouchDB. Since CouchDB does not allow any repeated document ID in its database, duplicated tweets were avoided directly. The harvester program uses two external libraries, namely Twitter 4j and couch4j. The former is used to invoke the Twitter Streaming API to harvest tweets, while the latter is responsible for checking the availability of CouchDB and saving the collected data. In addition to the Streaming API for harvesting real-time tweets, Twitter provides a RESTful API to search for recent tweets. There are two main approaches to access historical tweets to supplement recent tweets as follows.

3.2.3 *Cloud-based data processing*

After harvesting the Twitter data it is necessary to process the tweets to generate useful results. For this purpose we have used the MapReduce functions of CouchDB. CouchDB is an Apache open-source database, which unlike relational database management systems stores the data in the form of independent documents with each document identified with a unique ID.

For further analysis, non-English text and non-ASCII letters were filtered from the tweet content. The latitude and longitude of the tweets were used as keys and the user Tweet id as the value. The resultant dataset used for the experiments comprised more than 400 locations per user. The tweets were harvested from Miami between the time periods of April 25, 2015, to May 15, 2015. The total

number of tweets combining the data from the Search and Stream APIs after removing duplicates was 1,301,603. After preprocessing, the final dataset was composed of 308,264 locations from 1324 users. These data were saved with the following structure:

UserId|*PointID*|*Longitude*|*Latitude*|*Date*

3.3 Overview of the method

The software architecture used to support the explorations of location privacy of Twitter data is shown in Fig. 2. This architecture supports data collection (through the Twitter Search API although the Twitter Streaming API could also be used), data preparation and the associated methods required to perform ε-differential privacy and GLI pattern mining and associated analysis.

Differential privacy concepts were introduced in the previous sections. In this section, the methods used to generate a differential privacy-driven sanitization database from raw geo-located Twitter data are presented. This is achieved in two steps. The first step is to decompose spatial location regions by optimal quad-trees using differential privacy mechanisms. Following this, clustering of intersecting areas to find GLIs with perturbing outputs is undertaken to support differential privacy for locations, as shown in Fig. 3.

3.4 Differential privacy-based spatial decomposition

The classical solution to ensure differential privacy for spatial point datasets is to decompose the spatial space and then publish statistics on the points within each region in a differential privacy-preserving way. Users can get obfuscated knowledge of locations by intersecting the query regions with the split areas. The method to build

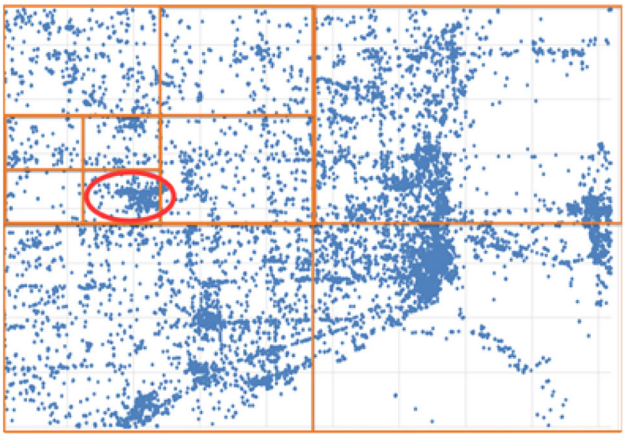
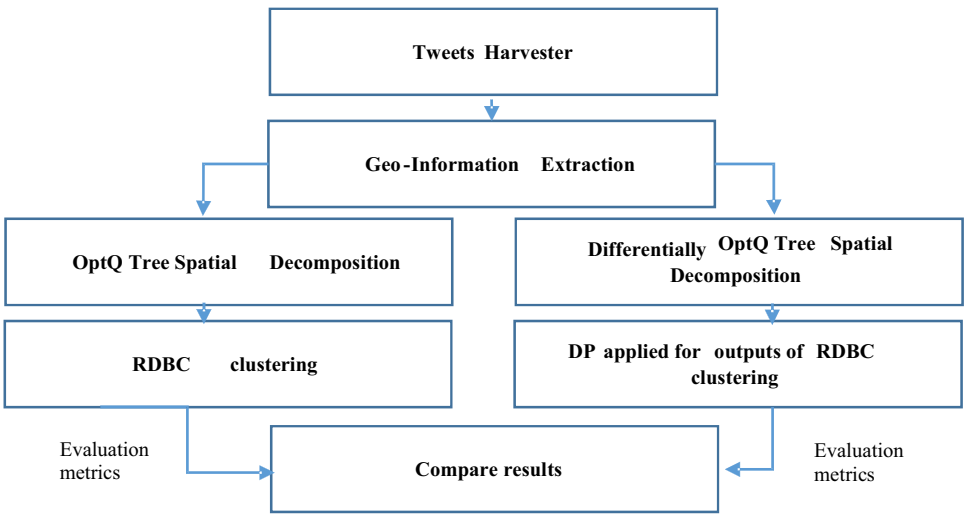


Fig. 3 Spatial decomposition sketch map (points in the *red circle* will be used for extracting the GLIs)

differential privacy spatial decomposition can be divided into adding noise to counts and index structures satisfying differential privacy. The purpose of spatial decomposition is to divide a global task into several local subtasks. Local sensitivity Δf_{local} is required in this situation (Eq. (4) previously). This approach can guarantee a better output location accuracy at a fixed differential privacy level since lower localized sensitivity results in lower σ for the Laplace noise mechanism Cormode et al. (2012).

There are two approaches that can be adapted to decompose (split) space: data-dependent and data-independent splitting. KD-tree is a data-dependent technique based on the distribution of points Ali et al. (2010), while quad-tree is a data-independent approach. A quad-tree-based spatial decomposition was adopted here to create sets of locations that group points within a certain area from the leaf of the quad-tree. As this can lead to privacy leaking when performing a non-perturbed spatial decomposition whereby attackers can retrieve the exact count of the points

Fig. 2 System architecture



within an area by simply comparing the dimension of the sub-region, the next step is to perturb the count of the sub-regions to protect the differential privacy of the count query outputs. This can be achieved by recalling the differential privacy idea that an attacker cannot guess if a particular point is or is not inside the database and if so, how many points fall within a certain area. Adaptive privacy budget strategy is used to achieve a more accurate decomposition. Algorithm 1 [OptQ-SDDP] is used to achieve differential privacy of the space decomposition; namely, some areas that should be split are kept whole, while others that should be kept whole are split.

There are various approaches to allocate privacy distributions across the tree including uniform distributions and geometric distributions. H is defined as the height of a tree; hence, the levels of the tree range from 0 to H. According to parallel composition, the privacy of all nodes on level i is ε_i. According to sequential composition and parallel composition, the sum of privacy across all levels should be ε, namely $\sum_{i=0}^{H} \varepsilon_i = \varepsilon$. To optimize the result of differential decomposition, an error measure method is introduced as follows: Let q represent any query and o' be the output of the query qp over the privacy tree. When the mean of Laplace distribution is 0, o' can be adapted as an unbiased estimator of the true output o. The variance of o', namely $V(o')$, can be represented as an indicator of error, namely Error $(q) = V(o')$.

The variance of the Laplace distribution Lap(ε_i), namely $V(\text{Lap}(\varepsilon_i))$, is $2/\varepsilon_i^2$. Let n_i denote the number of nodes contributed to q at level i of the quad-tree; hence, the n_i is 4^{h-i} if the quad-tree is full, in which the root is the level h and leaf is at level 0. For 2-dimensional quad-tree decomposition, let $n_i = 4^{H-i}$ when q includes the maximum (upper limit) number of counts at each level for all quad-tree count queries. Let $n(q)$ denote the number of nodes that contribute counts to q. For instance, as shown in Fig. 4, one query q is used to calculate the count of points in c2, c3, b2, c9, c10 and c14. Thus, the n_i in the different levels is 0, 1, 4, respectively, and $n(q) = n(q) = \sum_{i=0}^{H} n_i = 0 + 1 + 4$ in this instance. Consequently, $n(q) = \sum_{i=0}^{H} n_i \le \sum_{i=0}^{H} 4^{H-i} = \frac{1}{3}(4^{H+1} - 1)$, whose time complexity is $O(4^H)$. Since every node is independent from one another and every node at the same level of the tree has the same privacy value (and the same error) according to parallel composition in Sect. 3.1.1, it can thus be deduced that $E(q) = \sum_{i=0}^{H} 2n_i/\varepsilon_i^2$.

Note that the standard method (de Berg et al. 2008) to execute noise range queries is as follows: from the root to all nodes N whose rectangle is intersected by q. When q contains a whole node N, add the noisy count to the answer q_p; if not, traverse the child nodes of N until the leaves are reached. If leaf A intersects q but is not included in q, the uniformity assumption is adapted to determine that the noisy count can be added to q_p.

For uniform distributed privacy strategies, let $\varepsilon_i = \in /(H+1)$ be used for noise counts in trees. This approach has lower accuracy that seriously affects the next step and the whole accuracy. Here $E(q) = \sum_{i=0}^{H} 2n_i/\varepsilon_i^2 = \frac{2(H+1)^2}{\varepsilon^2} \sum_{i=0}^{H} n_i \le \frac{2}{3\varepsilon^2}(H+1)^2(4^{H+1} - 1)$.

For geometric distributed privacy strategies, accuracy can be significantly increased by a non-uniform privacy distribution strategy. Specifically, the following optimization problem is adapted to minimize the upper bound.

$$\text{Min} \sum_{i=0}^{H} 4^{H-i}/\varepsilon_i^2.$$

$$\text{Subject To} \sum_{i=0}^{H} \varepsilon_i = \varepsilon.$$

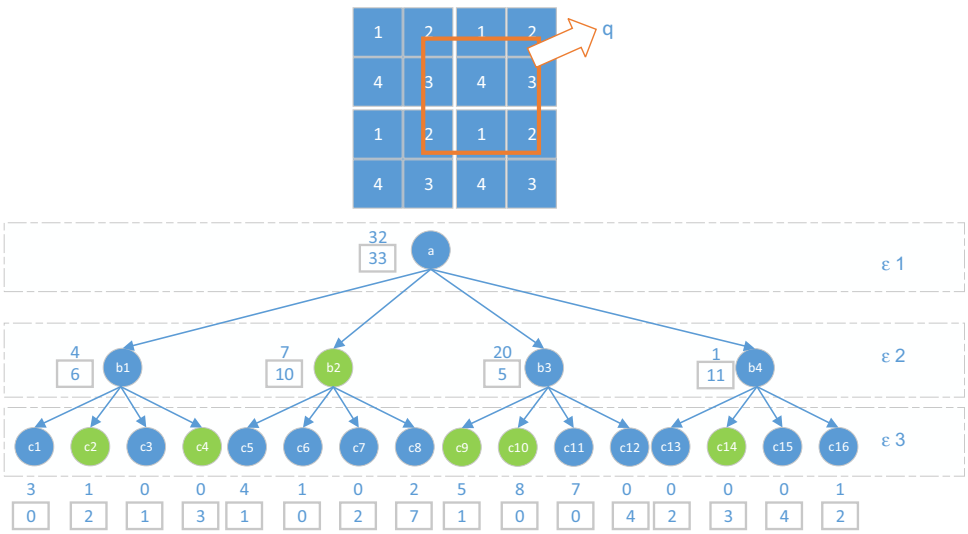


Fig. 4 Private quad-tree (counts in *rectangles* are with added noise and to be released)

An upper bound for $E(q)$ is

$$\frac{2\left(4^{\frac{H+1}{3}}-1\right)^3}{\varepsilon^2\left(\sqrt[3]{4}-1\right)^3}$$

Then $\varepsilon_i = 4^{(H-i)/3}\varepsilon\frac{\sqrt[3]{4}-1}{4^{(H+1)/3}-1}$.

Proof According to the Cauchy–Schwarz inequality, there is:

$$\left(\sum_{i=0}^{H}\varepsilon_i\right)\left(\sum_{i=0}^{H}\frac{4^{H-i}}{\varepsilon_i^2}\right) \geq \left(\sum_{i=0}^{H}\sqrt{\varepsilon_i 4^{H-i}/\varepsilon_i^2}\right)^2$$

This equality is obtained for all i only when $\varepsilon_i = C4^{H-i}/\varepsilon_i^2$, namely $\varepsilon_i = \sqrt[3]{C}4^{(H-i)/3}$ is attained (C is *constant*). According to $\sum_{i=0}^{H}\varepsilon_i = \varepsilon$, there is $\sqrt[3]{C} = \frac{\varepsilon\left(\sqrt[3]{4}-1\right)}{4^{(H+1)/3}-1}$. Hence, $\varepsilon_i = 4^{(H-i)/3}\varepsilon\frac{\sqrt[3]{4}-1}{4^{(H+1)/3}-1}$, and $\mathrm{E(q)} = \sum_{i=0}^{H}2n_i/\varepsilon_i^2 \leq 2\sum_{i=0}^{H}\frac{4^{H-i}}{\varepsilon_i^2} = 2\sum_{i=0}^{H}\frac{4^{H-i}}{(4^{(H-i)/3}\varepsilon\frac{\sqrt[3]{4}-1}{4^{(H+1)/3}-1})^2} = 2\frac{\left(4^{\frac{H+1}{3}}-1\right)^3}{\varepsilon^2\left(\sqrt[3]{4}-1\right)^3}$.

Hence, the upper bound is $2\frac{\left(4^{\frac{H+1}{3}}-1\right)^3}{\varepsilon^2\left(\sqrt[3]{4}-1\right)^3}$.

The goal is to minimize the resulting query errors. The worst error case is when q is a query that includes the maximum (upper limit) number of counts at each level, namely, $n_i = 8 \times 2^{H-i}$, as shown in Fig. 5, the worst error in the uniform privacy case is $E_{\mathrm{uni}}(q) = \frac{2}{3\varepsilon^2}(H+1)^2(4^{H+1}-1)$ and that of geometric privacy case is $2\frac{\left(4^{\frac{H+1}{3}}-1\right)^3}{\varepsilon^2\left(\sqrt[3]{4}-1\right)^3}$ with changes with the height of the tree. As seen, uniform privacy errors increase far more rapidly than geometric privacy errors.

The input of Algorithm 1 is a set S of points, e.g., pairs of coordinates with time stamps and a user ID, a spatial region R that is used for spatial decomposition, a maximum height H of the quad-tree and a threshold T, namely the minimum leaf size that is used to stop the recursion of the algorithm when the count of points in a sub-region falls below L, and an upper bound used for perturbed counts in a returned partition which is set to be $T = 3L$. The output is a set of spatial partitions P and a set S_p of points used for the corresponding partitions in P. The algorithm executes a noisy count of the current area points, namely *Count-WithNoise*, based on the local sensitivity corresponding to the current region, and compares it with the threshold L to determine whether it is necessary to keep splitting the area or to stop. The output of this algorithm contains both points and the corresponding user ID. Note that the maximum height H of the quad-tree is 8.

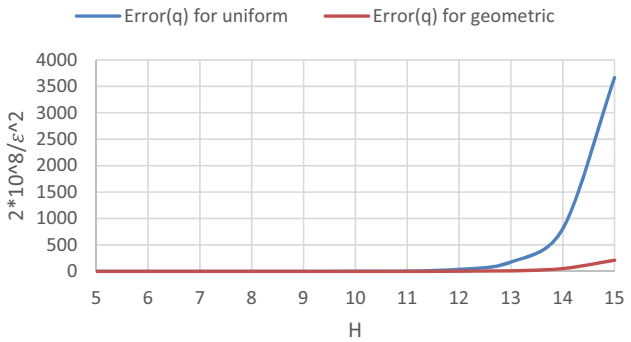


Fig. 5 Worst case uniform and geometric noise Err(q)

The upper bound for perturbed point counts can be set as $3*L$. As a result, the count sensitivity of the optimal quad-tree decomposition is given by $\Delta f_1 = 3*L$.

The Laplace noise σ in Lap(σ) is given by:

$$\sigma_1 = \frac{f_1}{\varepsilon_1} \tag{5}$$

Here ε_1 is the privacy budget distributed to the first step according to the space decomposition.

Algorithm 1. Optimal quad-tree spatial decomposition with differential privacy (OptQ-SDDP)

Variables: $P = \{\}$; Sp = {}; $H = 8$; $T = 3L$
OptQ -SDDP (S, R, T)
0: Obtain ε_i according to geometric privacy budget strategy
1: CountWithNoise = $|S| + \mathrm{Lap}(\Delta f/\varepsilon_i)$;
2: **if** $h > 8$ then
3: $P = \mathrm{P}\cup\{R\}$; Sp = Sp$\cup\{S\}$;
4: return
5: **else if** CountWithNoise $< L$ then
6: $P = \mathrm{P}\cup\{R\}$; Sp = Sp$\cup\{S\}$;
7: return
8: **else**
9: Split spatial region R into 4 equal quadrants
10: OptQ -SDDP (S{q1}; Rn{q1}; T);
11: OptQ -SDDP (S{q2}; Rn{q2}; T);
12: OptQ -SDDP (S{q3}; Rn{q3}; T);
13: OptQ -SDDP (S{q4}; Rn{q4}; T);
14: **end if**
15: **return**

3.5 Extracting GLIs with differential privacy guarantees

The classical solution to ensure differential privacy for spatial point datasets is to decompose the spatial space and then publish statistics on the points within each region from 4.3. To extract differential privacy GLIs, we use a density-based clustering algorithm (DBSCAN). A recursive density-based clustering algorithm (RDBC) is extended from DBSCAN (density-based spatial clustering of applications with noise). The advantages of RDBC are as follows:

- the number of clusters need not be specified;
- it can be used to find arbitrarily shaped clusters;
- it is robust to outliers (and hence to noise);
- changing the parameters (*Eps* and *MinPts*) intelligently during the recursively process ensures it is insensitive to the order of points;
- the identification of core points is performed separately from that of clustering individual data points

RDBC has further improvements to DBSCAN. RDBC calls DBSCAN with different distance thresholds ε and density threshold *MinPts* and returns the result when the number of clusters is appropriate. When abstracting, these core points can be regarded as clustering centers. Hence, the input parameters are used in RDBC, namely different values of ε and Mpts identify this core point set, CSet. Only after an appropriate CSet is determined, the core points are clustered, and the remaining data points are then assigned to clusters according to their proximity to a particular cluster (Su et al. 2001).

Algorithm 2. Extracting GLIs with differential privacy algorithm

1: Set initial values Eps = Eps_1 and Mpts = $Mpts_1$; $G = \{\}$; Cgp = {}; CT' = 0; CC' = (0,0); $M = \{\}$, Mc_j is a set of points in M

2: **for** $i = 1$ to |Sp| **do**

3: RDBC (Eps_1, Mpts, S_i)

4:Use Eps and Mpts to get the core points set CSet

5:**if** size (CSet) > size (DataSet)/2//Stopping criteria is met.

6:DBSCAN (DataSet, Eps, Mpts);

7:**else**//Continue to abstract core points;

8:Eps = Eps/2; Mpts = Mpts/4

9:RDBC (Eps, Mpts, CSet);//Collect all other points in around clusters

11: **end if**

12: **end for**

11: **for** $i = 1$ to |Sp| **do**

12: **for** $j = 1$ to |M| **do**

13: CT' = | Mc_i| + Lap(σ_{ct}^j);

14: **if** CT' > ic then

15: Centroid $CC_i = \frac{\sum_{k=1}^{|Mc_j|}(x_k, y_k)}{|Mc_j|}$

16: CC' = NoisyLap(σ^j)(CC_j)

17: G = G∪{CC'};

18: Cgp = Cgp∪{CT'};

19: **end if**

20: CC' = (0, 0), CT' = 0;

21: **end for**

22: **end for**

Algorithm 2 [DPGLIE-RDBC] is used to extract GLIs with differential privacy guarantees based on RDBC. As seen from Algorithm 2, the input variables are a set of location data subsets obtained by the previous step, threshold *ic*, initial *Eps* and *MinPts* for RDBC. It is noted that *Eps* and *MinPts* can be changed intelligently during the recursive loop. Lap(σ_{ct}) is used for perturbing the counts of each cluster *Cj* extracted by RDBC, and Lap(σ_{cc}) is used to perturb the centroid of each cluster C_j extracted by RDBC. If the perturbed count CT' is greater than the threshold *ic*, then the region C_j is marked as a GLI. The centroid of C_j is used for the next step of the privacy evaluation metrics. The output of this algorithm is G—the set of privacy-preserved GLIs given as the region centroids, Cgp, i.e., the set of privacy-preserved counts of points.

The count sensitivity and the centroid sensitivity for the cluster C_j are given as follows. The count sensitivity Δf_{ct}^j is defined as MAX(NUM$_{individual}$(points)), $\forall individual \in C_j$. So $\sigma_{ct}^j = \Delta f_{ct}^j/\varepsilon_{ct}$, where ε_{ct} is the privacy distribution in the counting points step. The centroid sensitivity Δf_{ct}^j is defined as MAX(distance (p_i, p_j))/2 $\forall$ p_i, $p_j \in C_j$. So $\sigma_{ct}^j = \Delta f_{cc}^j/\varepsilon_{cc}$, where ε_{cc} is the privacy level of the counting centroid step.

This algorithm contains a loop where the core points are regarded as points in a space on which to cluster. The stop condition is when nearly half the points that remain are core points. At this point, the algorithm will begin a gathering process to gather the rest of the points around the core points found in clusters with radius value Eps_2.

Note that the method that NoisyLap(σ) perturbs a real location coordinate $l_r(x_r, y_r)$ to a perturbed location coordinate $l_p(x_p, y_p)$ was introduced by Yonghui et al. (2006). Accordingly, our perturbing approach is achieved by using a Laplace distribution with scale $\sigma > 0$ to perturb a location $l_r(x_r, y_r)$ such that:

$$\begin{aligned} \Pr(x_r \to x_p) &= \frac{1}{2\sigma} e^{\frac{|x_r - x_p|}{\sigma}} \\ \Pr(y_r \to y_p) &= \frac{1}{2\sigma} e^{\frac{|y_r - y_p|}{\sigma}} \end{aligned} \tag{6}$$

In (6), σ is set at $(\max_n x_n - \min_n x_n)/\varepsilon_{cc}$ to generate x_p and set at $(\max_n y_n - \min_n y_n)/\varepsilon_{cc}$ to generate y_p. It should be noted that this approach for achieving a Laplace noise mechanism is to perturb c to such $c - \sigma \mathrm{Sgn}\ (q)\ \ln(1 - 2|q|)$, where q is a random value drawn from a uniform distribution between [−0.5, 0.5], and Sgn is a function that distributes the perturbation around c.

Figure 6 illustrates the extracting GLIs with differential privacy guarantees. The real GLI can be extracted from each cluster centroid (round shapes), followed by sanitized the centroid by adding random noisy drawn from Laplace distribution to provide differential privacy guarantees, as the perturbed GLI (triangle shapes).

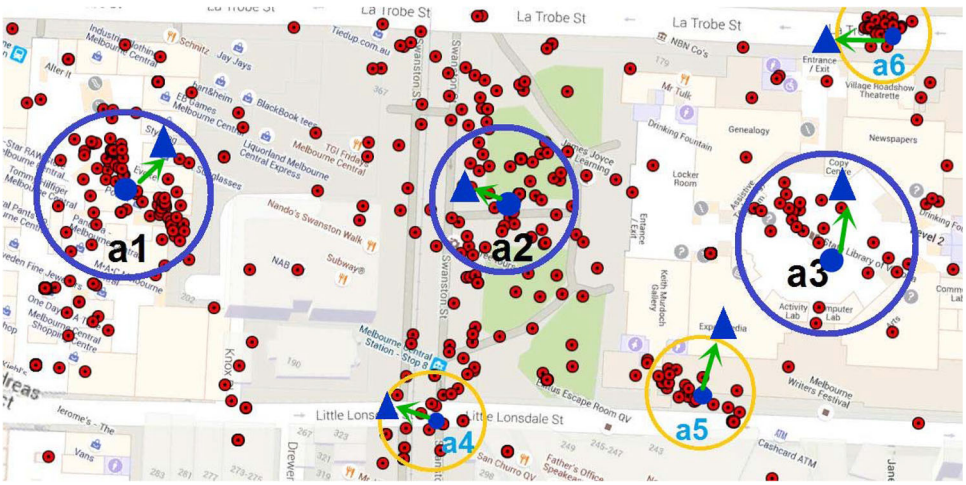


Fig. 6 Visualization for extracting the GLIs

3.6 Privacy level distribution

The two important properties in Sect. 2 prove that ε-differential privacy guarantees can be implemented by performing a sequence of differential privacy mechanisms. The privacy leak level ε can be composed of $\varepsilon = \varepsilon_1 + \varepsilon_{ct} + \varepsilon_{cc}$ where ε_1 is the privacy leakage level used for the optimal quad-tree spatial decomposition, ε_{ct} is the privacy leakage level used for perturbing the count of numbers of points in each cluster, and ε_{cc} is the privacy leakage level used for perturbing the count of centroids comprising each cluster, for instance, if the database must guarantee a maximum privacy leak level of $\varepsilon = 0.8$. One can subdivide the ε by $0.8 = 0.3 + 0.3 + 0.2$. Factoring in the optimal quad-tree decomposition h, it can be shown that $\varepsilon_1 = \sum_{i=1}^{h} \varepsilon_{qt}$. Combined with sequential composition, the overall privacy leak level can be given as $\varepsilon = \sum_{i=1}^{h} \varepsilon_{qt} + \varepsilon_{et} + \varepsilon_{cc}$.

4 Evaluation metrics

In this section, the evaluation metrics used to measure the applicability of the approach described are presented. Specifically, we evaluate the utility and privacy features of the differential privacy location pattern-mining method to discover GLIs. These evaluation metrics contain the inferred number of actual GLIs, the Euclidean distance between actual GLIs and location privacy-enabled GLIs, the count difference of points in the intersection of real regions and privacy-preserving regions, as well as the number of similar neighborhoods surrounding real GLIs and location privacy-enabled GLIs.

4.1 Metrics for measuring utility

To measure the utility of the privacy-preserving mechanisms, we take the view of obfuscated data users (ODUs) who want to draw knowledge from perturbed locations by sending queries and running the DPGLIE-RDBC algorithm. Metrics for measuring utility are given for assessment of the distortion of obfuscated GLIs inferred by the ODUs compared to the actual GLIs. The notations used in this section are listed in Table 1.

Let IS be the set of intersections of the sets SR and SP where SR is the set of regions with real points and SP is the set of regions with privacy-preserved points. Note that the intersection between the two regions' intersection is not empty (Primault et al. 2014).

The first step is to find the corresponding real GLIs for each perturbed GLI discovered by the DPGLIE-RDBC algorithm. In this situation, we calculate the nearest real

Table 1 Notions used in evaluation metrics

Name	Description
IS	IS is the set of intersections of the sets SR and SP
SR	The set of regions with real points
SP	The set of regions with privacy-preserved points
CTr	$CTr = \{c_{1r}, c_{2r}, \ldots, c_{\|IS\|,}\}$
C_{ir}	The count of points in each region in SR
CTp	$CTp = \{c_{1p}, c_{2p}, \ldots, c_{\|IS\|,}\}$
C_{ip}	The count of points in each region in SP
CCr	$CCr = \{cc_{1r}, cc_{2r}, \ldots, cc_{\|IS\|,}\}$
cc_{ir}	The centroid (Xcc_{ir}, Ycc_{ir}) of points in each region in SR
CCp	$CCp = \{cc_{1p}, cc_{2p}, \ldots, cc_{\|IS\|,}\}$
cc_{ip}	The centroid (Xcc_{ip}, Ycc_{ip}) of points in each region in SP

GLI to the corresponding perturbed GLIs, where these GLIs have been reduced to their centroids.

The first metric is *recall*, namely, the number of real GLIs inferred by the ODUs. The *recall* can be defined as follows:

$$\text{Recall} = \frac{(\text{count of GLIs that have more than one GLI in the IS})}{(\text{count of GLIs in the SR})} \tag{7}$$

As we can see from the definition of *recall*, this can be used to assess the percentage of GLIs that have been discovered from the set of real GLIs by ODUs.

Although *recall* can reflect the percentage of discovered GLIs, the distortion of those GLIs is not assessed. Hence, the function *GeographicDistance* uses geographic coordinates to calculate the Euclidean distance between real GLIs and perturbed (obfuscated) ones to represent the utility of the privacy-preserving solution. This can be formulized as follows:

$$\text{Geographic Distance} = \text{dist}(g_r \rightarrow g_o) \tag{8}$$

Specifically, the cumulative distance distribution is adapted, e.g., the ratio of distances of discovered GLIs to their corresponding obfuscated GLIs and to all discovered GLIs.

4.2 Metrics for measuring precision

GLIs are typically used to provide location-based services for user, e.g., finding nearby hospitals, hotels and restaurants. Twitter applications such as "Nearby" allow users to find their friends' tweets in a given vicinity. Hence, we assess the precision of our approach in measuring distances to GLIs. Specifically, we use the nearest neighbors search service provided by location-based services to discover the top 20 shops around given regions of target city (considered as centroids in the IS) and count the number of similar shops between real GLI centroids and perturbed ones. Specifically, we calculate the precision as the count of the intersection of these two sets of shops (out of 20).

5 Experimental results

In this section, our objective is to evaluate the privacy and utility of the differential location pattern-mining approach as described in Sect. 4 in terms of the metrics introduced in Sect. 5 and apply this approach for traffic information alerting. As noted our implementation was performed on virtual machines offered through the NeCTAR Research Cloud. The implementation itself was done in Java and Python.

5.1 Datasets

We used the tweet location datasets as described in Sect. 3 to implement the experiments. This dataset contained 308,264 geospatially tagged tweets from Miami-based (geo-located) Tweeters with bounding box SW: [−80.320773, 25.711586], NE: [−80.136924, 25.864451].

For each of these, the (latitude, longitude) coordinate values were expressed in (x, y) rectangular coordinates with (0, 0) respecting (−80.320773, 25.711586) in the bottom left (the coordinates for Miami). The distance between each coordinate was calculated based on the Euclidean distance.

5.2 Extracting GLIs from real locations

We used the optimal quad-tree spatial decomposition method to split the region to smaller sub-regions, in which the threshold value T was set to 500. Following this RDBC was used to cluster each sub-region. Finally, 90 GLIs were identified containing some notable GLIs. Based on this, we set a count of points in each sub-region (CTr) and a set of centroids of each sub-region (CCr).

5.3 Extracting GLIs from Twitter with privacy-preserving mechanisms

We set the threshold value T of the spatial decomposition (DP-optimal quad-tree) to 500, so the upper bound of points in a given region is 1500. Other parameters were set as shown in Table 2.

As the privacy-preserving mechanism is based on a randomized approach, the results obtained are not deterministic. Therefore, we ran the experiment 20 times to obtain 20 independently obfuscated datasets, and the final results represent the mean value of these outputs. Note that the experiments were performed on three classical differential privacy leakage levels obtained by experiments and are shown in Table 3.

From this we identified 102 obfuscated GLIs. As a result, we obtained a set of count of points in each sub-region (CTp) and a set of centroids for each sub-region (CCp).

Table 2 Parameters setting

Name	Value
T	500
MinPts	50
Eps	0.1
ic	50

Table 3 Different privacy leakage levels for whole

Level of ε	Distribution of ε		
	ε_1	ε_2	ε_3
Strong	0.1	0.01	0.01
Normal	1	0.5	0.5
Weak	5	1	1

5.3.1 Utility evaluation

Recall In this part, the most important task was to find the threshold that can be used to declare whether the real GLI was discovered or not. An optimal threshold can be used to ensure a high recall and associated low distance among GLIs. The way we address this is to set the minimum Euclidean distance between the real location and the obfuscated one at which the recall is higher than 70% of the threshold. We have assessed the recall of differentially privacy-based optimal quad-tree spatial decomposition algorithm (OptQ-SDDP) and RDBC with differential privacy protection levels (DPGLIE-RDBC), respectively. The results of recall of whole are shown in Table 4 and Fig. 7. It is clear that the recall of whole rate increases as ε becomes larger. That is to say, privacy and precision are trade-offs, i.e., the higher the degree of privacy protection, the fewer the GLIs will be identified. Using the method described above, the thresholds are also determined as shown in Table 5.

Table 6 shows the different privacy leakage levels of the DP-RDBC. As can be seen from Fig. 8, the larger the $\varepsilon_2 + \varepsilon_3$ are, the larger the recall will be. As above, if the degree of privacy protection is higher, fewer GLIs will be found. The thresholds are shown in Table 7.

Table 4 Recall for different privacy leakage settings

Level of ε	Recall of whole (%)
Strong	70.34
Normal	71.58
Weak	75.69

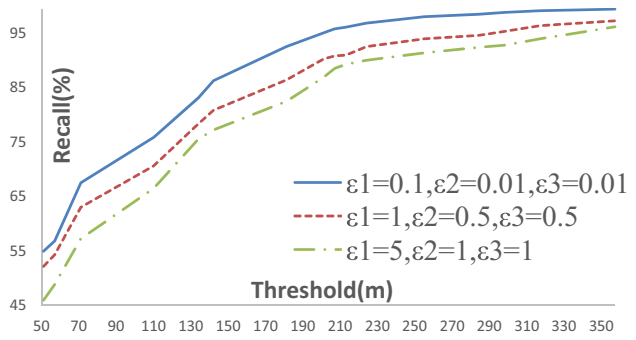


Fig. 7 Recall for different privacy levels

Table 5 Threshold distance of whole

Level of ε	Threshold of whole
Strong	101
Normal	105
Weak	117

Table 6 Different privacy leakage levels for DPGLIE-RDBC

Level of ε	Distribution of ε	
	ε_2	ε_3
Strong	0.01	0.01
Normal	0.5	0.5
Weak	1	1

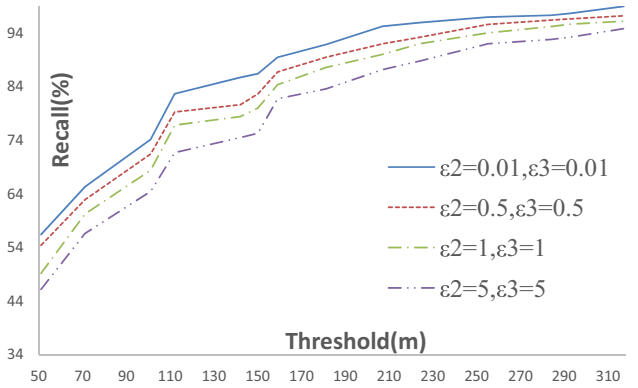


Fig. 8 Recall for DPGLIE-RDBC levels

Table 7 Threshold distance of DPGLIE-RDBC

Level of ε	Threshold of DPGLIE-RDBC
Strong	102
Normal	106
Weak	116

Regarding the relationship between privacy and precision in DP-QT, different ε_3 are picked to decompose the space using the DP-optimal quad-tree algorithm and used to evaluate the recall of DP-QT. The results of recall of DP-QT are shown in Table 8 and Fig. 9. Similarly, privacy and precision are trade-offs in the DP-QT as shown.

Geographic distance Geographic distance between real GLIs and corresponding obfuscated ones across all users for different values of privacy leakage level is shown in Fig. 10. This shows the percentage of GLIs that resulted in the perturbed points being generated within thresholds determined from the real GLIs. Specifically, when the ε is at the smallest level, i.e., where strong privacy protection

Table 8 Different privacy leakage levels for optimal quad-tree

Level of ε	Distribution of ε ε_1
Strong	0.01
Normal	1
Weak	5

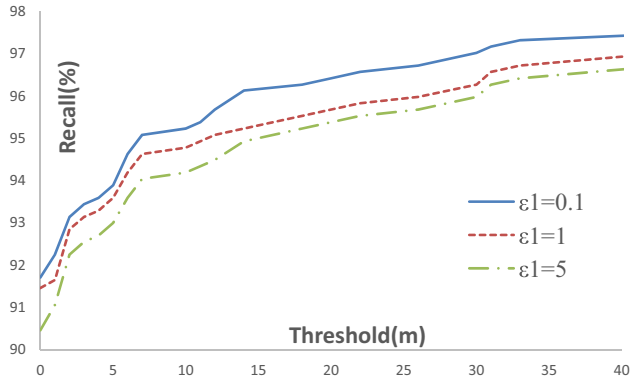


Fig. 9 Recall for optimal quad-tree levels

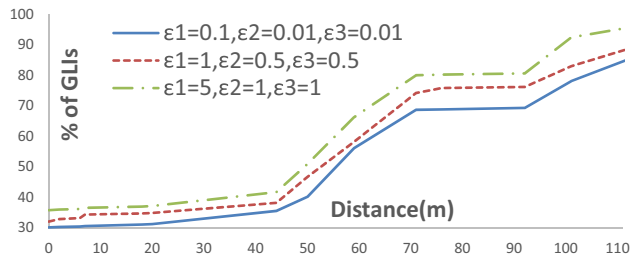


Fig. 10 Geographic distance distribution

strength is demanded, only about 56% of GLIs are within 70 m, while it can reach 67% when ε is at the highest level.

5.3.2 Precision evaluation

To assess the precision of our approach, we explore a typical query by a location-based service, such as: "find all shops within 500 meters of my current location." To answer this, we consider the percentage of similar results between real centroids and obfuscated centroids, respectively. Figure 11 shows the percentage of GLIs that have a similarity of more than 10% at different privacy of 15, 45 and 70%, respectively, i.e., when ε is smaller, stronger privacy protection arises and hence the similarity will increase.

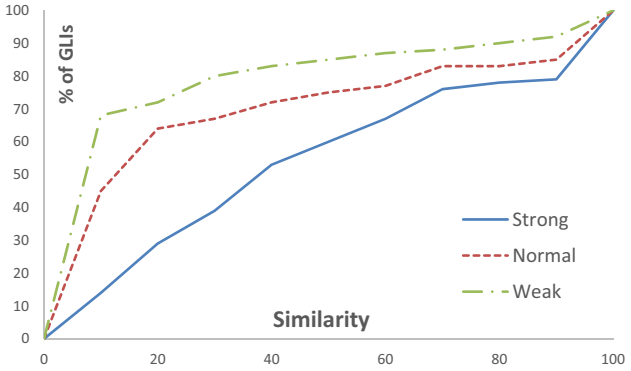


Fig. 11 Precision evaluation

5.4 Privacy-preserving traffic analysis

One key area of application of Twitter is real-time information on transport. Tweets about traffic conditions such as traffic congestion or traffic accidents provide near real-time traffic information that is useful for travelers and could allow them to take alternative routes or make other travel plans. As Twitter is becoming increasingly popular and has provided location-based services like "Nearby," more and more real-time road traffic information with users' identification can be collected from actual users traveling on the roads. However, users' privacy information such as time-stamped locations and movements is also given. Hence, privacy of the individual location and the identity of the user are key to protect when mining the location pattern. In this case, we consider how to aggregate GLIs from related tweets with geographic coordinates while protecting the privacy of the users' locations.

We collected 74,519 traffic-related tweets with location information harvested between March and May 2015 across Miami as shown in Fig. 12, filtered using semantic analysis. In this figure, the location of the tweets (and hence Tweeter) is shown by a set of dots and then visualized in aggregate level through a heat map. The first step is to spatially separate these locations through the optimal quad-tree algorithm and then aggregate them by the RDBC algorithm to obtain GLIs as the round shapes as shown in Fig. 13 left. Note that there are tweets with location information that may not be associated with traffic events, i.e., the work did not tackle natural language processing or more advanced semantic analysis of the tweets.

The next step is to decompose the spatial location regions by optimal quad-tree incorporating differential privacy mechanisms. Following this, clustering intersecting of areas to find GLIs with perturbed outputs is undertaken to support differential privacy of location information. This results in obfuscated GLIs represented by the triangle shapes, as shown in Fig. 13 right.

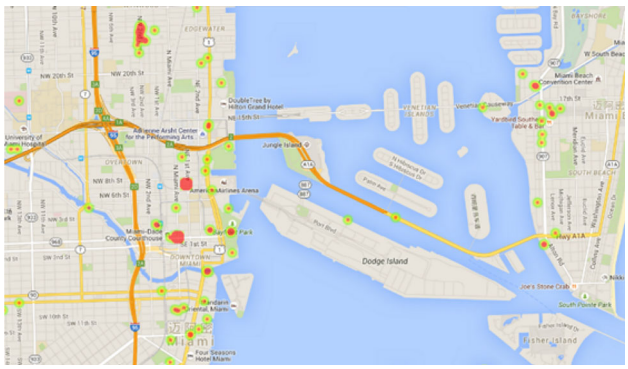


Fig. 12 Traffic accident-related tweet geographic locations

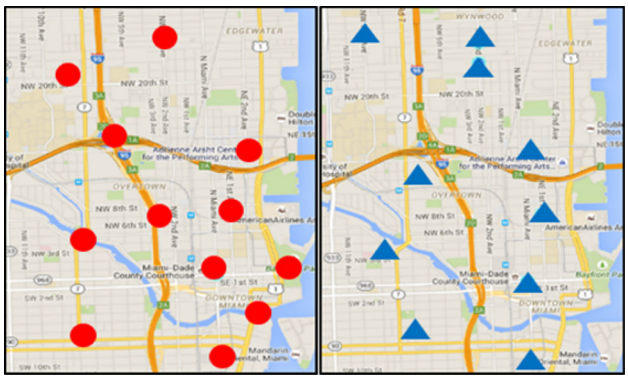


Fig. 13 GLIs of real geographic locations data

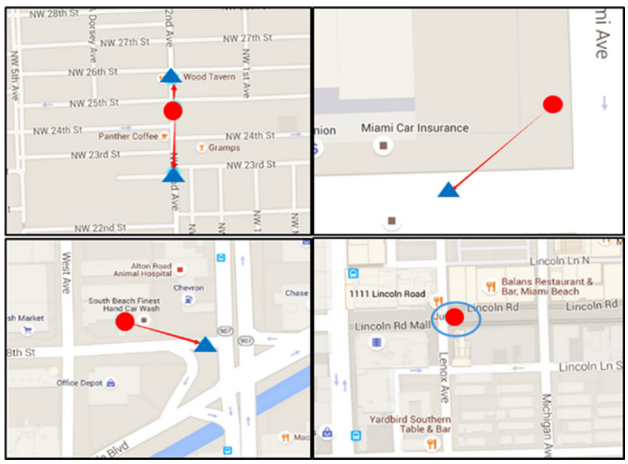


Fig. 14 Change of obfuscated GLIs compared to real ones

Figure 14 displays the change of the obfuscated GLIs (triangle shapes) compared to the real ones (round shapes). As we can see in Fig. 14, a real GLI can have zero, one or many obfuscated GLIs according to differential privacy-preserving levels. Thus, when users want traffic information displayed by this method, they can identify areas with a concentration of traffic-related tweets and hence avoid these areas and potentially pick another route as shown. By analyzing tweets collected over long periods, we can find areas where traffic congestion or traffic accidents are more likely to occur and alert drivers regarding congested roads with alternative routes recommended.

In addition, this method can not only protect a user's location privacy while efficiently ensuring the accuracy of the location-based service through differential privacy, but also protect the privacy of each individual user by adding noise to the statistical reports so that a user's tweets cannot significantly change the alert status.

6 Conclusion

In this paper, we explored adding differential privacy capabilities to Twitter data. Through the application of RDBC to cluster sub-regions split by differentially privacy optimal quad-tree spatial decomposition, we explored privacy of Geo-Locations of Interest (GLIs). We assessed this approach by comprehensive metrics covering both privacy and precision levels of Twitter data. We showed that privacy and precision are trade-offs, noting that differential privacy noise mechanisms are indeed an effective way to provide location privacy of Twitter data. As shown, the location precision will decrease when the privacy protection level increases. In the future, we will explore the impact of temporal information on user tweets and how to protect other interconnected information. We shall also explore algorithms that allow these methodologies to be used in the context of much larger datasets. For example, we have currently harvested over 40 Tb of Twitter data from across Australian on a range of topics and from a range of regions; the computational overheads of ensuring privacy in such circumstances become challenging. We shall also explore the practical realities of this work in a range of health projects where social media is required, e.g., national pandemic projects currently starting up at the University of Melbourne focused on emerging infectious diseases.

Acknowledgements We would like to thank the NeCTAR Research cloud for the (free) use of the cloud resources and the Melbourne eResearch Group for support on Twitter access, use and analysis. Figure 1 was produced as part of the Australian Urban Research Infrastructure Network (AURIN—www.aurin.org.au) project.

References

Abul O, Bonchi F, Nanni M (2008) Never walk alone: uncertainty for anonymity in moving objects databases. In: Data engineering, 2008. ICDE 2008. IEEE 24th international conference on, pp 376–385. doi:10.1109/icde.2008.4497446

Ali I, Kantarcioglu M, Ghinita G, Bertino E (2010) Private record matching using differential privacy. In: Manolescu I,

Spaccapietra S, Teubner J, Kitsuregawa M, Leger A, Naumann F, Ailamaki A, Ozcan F (eds) Proceedings of the 13th international conference on extending database technology (EDBT '10). ACM, New York, pp 123–134. doi:10.1145/1739041.1739059

Andrés ME, Bordenabe NE, Chatzikokolakis K, Palamidessi C (2013) Geo-indistinguishability: differential privacy for location-based systems. In: Proceedings of the 2013 ACM SIGSAC conference on computer & communications security (CCS '13). ACM, New York, pp 901–914. doi:10.1145/2508859.2516735

Arik F, Schuster A (2010) Data mining with differential privacy. In: Proceedings of the 16th ACM SIGKDD international conference on knowledge discovery and data mining (KDD '10). ACM, New York, pp 493–502. doi:10.1145/1835804.1835868

Ashwin M, Kifer D, Abowd JM, Gehrke J, Vilhuber L (2008) Privacy: theory meets practice on the map. In: Alonso G, Blakeley JA, Chen ALP (eds) Proceedings of the 24th international conference on data engineering, ICDE 2008, April 7–12, 2008, Cancún, México, pp. 277–286. IEEE

Changqing Z, Frankowski D, Ludford P, Shekhar S, Terveen L (2004) Discovering personal gazetteers: an interactive clustering approach. In: Proceedings of the 12th annual ACM international workshop on Geographic information systems (GIS '04). ACM, New York, pp 266–273. doi:10.1145/1032222.1032261

Chow C-Y, Mokbel MF, Aref WG (2009) Casper*: query processing for location services without compromising privacy. ACM Trans Database Syst. Article 24 (December 2009). doi:10.1145/1620585.1620591

Cormode G, Procopiuc C, Srivastava D, Shen E, Yu T (2012) Differentially private spatial decompositions. In: Data engineering (ICDE), 2012 IEEE 28th international conference on, 20–31. doi:10.1109/icde.2012.16

de Berg M, Cheong O, van Kreveld M, Overmars M (2008) Computational geometry: algorithms and applications. Springer, Berlin

Dewri Rinku (2012) Location privacy and attacker knowledge: Who are we fighting against? Lect Notes Inst Comput Sci Soc Inform Telecommun Eng. doi:10.1007/978-3-642-31909-9_6

Dwork C (2006) Differential privacy. In: Automata, languages and programming, ser. Lecture Notes in Computer Science. Springer, Berlin, vol 4052, pp 1–12. doi:10.1007/11787006_1

Dwork C, McSherry F, Nissim K, Smith A (2006) Calibrating noise to sensitivity in private data analysis. In: Proceeding of the 3rd conference on theory of cryptography, NY, pp 265–284. doi:10.1007/11681878_14

Hasan S, Zhan X, Ukkusuri SV (2013) Understanding urban human activity and mobility patterns using large-scale location-based data from online social media. In: Proceedings of the 2nd ACM SIGKDD international workshop on urban computing (UrbComp '13). ACM, New York, Article 6. doi:10.1145/2505821.2505823

Ho S-S, Ruan S (2011) Differential privacy for location pattern mining. In: Proceedings of the 4th ACM SIGSPATIAL international workshop on security and privacy in GIS and LBS (SPRINGL '11). ACM, New York, pp 17–24. doi:10.1145/2071880.2071884

Ho S-S, Ruan S (2013) Preserving privacy for interesting location pattern mining from trajectory data. Trans Data Priv 6(1):87–106

Hu H, Xu J, On ST, Du J, Ng JK-Y (2010) Privacy-aware location data publishing. ACM Trans Database Syst. Article 18 (July 2010). doi:10.1145/1806907.1806910

Jiang K, Shao D, Bressan S, Kister T, Tan K-L (2013) Publishing trajectories with differential privacy guarantees. In: Szalay A, Budavari T, Balazinska M, Meliou A, Sacan A (eds) Proceedings of the 25th international conference on scientific and statistical database management (SSDBM). ACM, New York, Article 12. doi:10.1145/2484838.2484846

Kido H, Yanagisawa Y, Satoh T (2005) Protection of location privacy using dummies for location-based services. In: Data engineering workshops, 2005. 21st international conference on (ICDEW'05), IEEE, pp 1248–1248. doi:10.1109/icde.2005.269

McSherry F (2009) Privacy integrated queries: an extensible platform for privacy-preserving data analysis. In: SIGMOD, 2009. doi:10.1145/1559845.1559850

Ninghui L, Li T, Venkatasubramanian S (2007) t-Closeness: privacy beyond k-anonymity and l-diversity. ICDE 7:106–115

Nissim K, Raskhodnikova S, Smith A (2007) Smooth sensitivity and sampling in private data analysis. In: Proceedings of the thirty-ninth annual ACM symposium on theory of computing (STOC '07). ACM, New York, pp 75–84. doi:10.1145/1250790.1250803

Primault V, Mokhtar SB, Lauradoux C, Brunie L (2014) Differentially private location privacy in practice. Dans mobile security technologies conference, San Jose, pp 1–10

Sadeh Norman, Hong J, Cranor L, Fette I, Kelley P, Prabaker M, Rao J (2009) Understanding and capturing people's privacy policies in a mobile social networking application. Pers Ubiquitous Comput 13(6):401–412. doi:10.1007/s00779-008-0214-3

Su Z, Yang Q, Zhang H, Xu X, Hu Y (2001) Correlation-based document clustering using web logs. In: Proceedings of the 34th Annual Hawaii international conference on system sciences (HICSS-34), 2001, vol 5, pp 5022–5028. doi:10.1109/hicss.2001.926536

Sweeney L (2002) k-anonymity: a model for protecting privacy. Int J Uncertain Fuzziness Knowl-Based Syst 10(05):557–570. doi:10.1142/s0218488502001648

Terrovitis M, Mamoulis N (2008) Privacy preservation in the publication of trajectories. In: Mobile data management, 2008. MDM'08. 9th international conference on, pp 65–72. doi:10.1109/mdm.2008.29

Xiao Y, Xiong L, Yuan C (2010) Differentially private data release through multidimensional partitioning. In: Proceedings of the secure data management, 7th VLDB workshop, Singapore, Sep. 2010, pp 150–168. doi:10.1007/978-3-642-15546-8_11

Xiao X, Wang G, Gehrke J (2011) Differential privacy via wavelet transforms. Knowl Data Eng IEEE Trans 23(8):1200–1214. doi:10.1109/icde.2010.5447831

Xue M, Kalnis P, Pung HK (2009) Location diversity: enhanced privacy protection in location based services. In: Location and context awareness. Springer, Berlin, pp 70–87. doi:10.1007/978-3-642-01721-6_5

Yu Z, Zhang L, Ma Z, Xie X, Ma W-Y (2011) Recommending friends and locations based on individual location history. ACM Trans Web, Article 5 (February 2011). doi:10.1145/1921591.1921596