Differential Privacy Project - Survey Paper
Applying Differential privacy on Big Data
Gunjan Batra

Abstract - In the past ten years, we have seen a revolutionary growth in the user generated data. With the increase in number of people, devices, and sensors connected with the digital networks, data can be collected and analyzed to understand many important phenomena. While trying to gain useful information from this data, at the same time, it is very important to preserve the differential privacy of the individuals in the data. The research presents a survey of five such approaches for privacy preserving data analytics to mine this data collected and derive meaningful conclusions from it.

1. Introduction

We live in an era of big data. There is a huge amount of data generated from various sensors and devices in the digital world. This instrumentation of the physical world has given us the opportunity to collect useful data, analyze the data and mine it to understand important phenomenon. We can analyze the data for public (For example: studying disease outbreaks) as well as commercial interests. (For e.g. developing recommender systems). Our goal is to perform this data analytics such that there is no further loss in the privacy of the participating user. And the patterns in the user generated data could not be used to re-identify the individuals. The approaches discussed further provide such practical framework for data analytics and provide a guarantee of differential privacy to the individuals contributing to the data set.
The research papers deal with the problem of applying Differential Privacy while querying huge statistical databases. They are all competing with each other and we see an increasing trend in the efficiency achieved in these papers in a chronological order.

2. Approaches Proposed

The following approaches are proposed in the five papers surveyed:
1. Calibrating Noise to Sensitivity in Private Data Analysis, 2006.
2. Differentially private aggregation of distributed time-series with transformation and encryption, 2010.
3. Practical differential privacy via grouping and smoothing, 2013.
4. A Practical Framework for Privacy-Preserving Data Analytics, 2015.
5. Differentially Private Publishing of High-dimensional Data Using Sensitivity Control, 2015

The first four approaches came one after another annually and showed improvement in their methodology with time. The fifth approach came after the third one and shows improvement over the first three.

2.1. Calibrating Noise to Sensitivity in Private Data Analysis

The first paper publishes in 2006, is the original work of Cynthia Dwork in which she describes the Laplace Mechanism [1] to guarantee Differential Privacy. To protect the privacy of a statistical database, being queried by a function f, the query answer is perturbed by adding random noise generated by Laplace distribution. The scale (standard deviation) of this Laplacian noise added is calibrated to the sensitivity of the function f (divided by Є). For example, we have a counting query of sensitivity 1, then (Є, 0)-differential privacy can be achieved for the query by addition of noise scaled to 1/ Є. This method also helps us in adding controlled noise to the functions with low sensitivity.

Query Sensitivity: According to the standard technique proposed by Dwork et al we can achieve differential privacy, by adding random noise to the query answers, where the noise distribution is carefully calibrated to the sensitivity of the query.
The calibration depends on the query *sensitivity*, i.e. the maximum amount the query answers can change given any change to a single user's data I_u. Sensitivity then measures the distance between the two Query answer vectors - Q(I) and Q(I'), using the L_1 distance metric, denoted as $|Q(I)-Q(I')|_1$ that measures the Manhattan distance $\sum_i|Q_i(I)-Q_i(I')|$
Definition - Let Q be any query sequence. For p ϵ {1,2}, the L_p sensitivity of Q, denoted $\Delta_p(Q)$ is The smallest number such that for all I and I' € nbrs(I),

$$|\mathbf{Q}(I) - \mathbf{Q}(I')|_p \leq \Delta_p(\mathbf{Q})$$

Laplace Perturbation Algorithm (LPA) - To ensure differential privacy in presence of a trusted server, [1] proposes the LPA algorithm that adds noise generated according to Laplace distribution to the query answers. Probability Distribution Function of Laplace Distribution: Pr(Lap (λ) =Z) = $\frac{1}{2\lambda}\ e^{-\epsilon|Z|/\lambda}$. Lap(λ) has mean 0 and variance 2.
Let Lap^n to be a vector of n independent Lap(λ) random variables. The LPA algorithm takes as input a query sequence Q and parameter λ controlling the Laplace noise. LPA perturbs the query answers, Q(I) by adding independent Laplace Noise Lap(λ) to each query answer in Q(I). It computes and outputs Q=Q(I) + $Lap^n()$. For ensuring Differential Privacy, λ is calibrated according to the L_1 sensitivity of Q.
Theorem - LPA (Q, λ) is Є-Differentially Private for λ = Δ_1 (Q)/Є

2.2. Differentially Private Aggregation of Distributed Time Series with Transformation and Encryption

Problem Description - The second paper by Rastogi et al in 2010, proposes an algorithm (PASTE) [2] for ensuring Differential Privacy for distributed time series data, specifically the data collected by individual users which is temporally correlated. For example, location traces in Traffic applications, web history from Web browsers, personal health data from Health-care applications, etc. They are referring to historical recurring queries, example, the average weight of all users

computed once every month of the year. For such data and queries, the above-mentioned Laplace technique [1] is not efficient.

Data Set – They use 3 evaluation data sets:

1. GPS: This data contains GPS trace of timestamped latitude and longitude points of discrete trips collected from 253 voluntary drivers in Seattle comprising of discrete trips. The trace covering about 13500 kilometers of driving was a part of Microsoft's Multiperson Local Survey. The median interval between recorded points on a trip is 6 seconds and 62 meters.
2. Weight: This is trace of daily body weights of 300 users collected for a period of 5 years
3. Traffic: This data included a 6-month long trace of traffic-volume and speed at about 30 intersection in San Antonio, Texas.

Methodology - PASTE uses FPA (Fourier Perturbation Algorithm) to perturb the Discrete Fourier Transform (DFT) of the query answers. This results in huge improvement of error for answering n queries (from Θ(n) to Θ(k) where k is number of Fourier coefficients that can reconstruct the n query answers). After this, to solve the problem of absent central server, PASTE uses the DLPA (Distributed Laplace Perturbation Algorithm) for adding noise in a distributed way. This scales very well with large number of users and reduces the computational load per user from O(U) to O (1), where U is the number of users. They perform experiments with real world datasets which show that PASTE improves accuracy of query answers by orders of magnitude compared with LPA [1] and also it scales very well with large number of users.

The two protocol stages used by PASTE for answering a sequence Q of n queries are discussed below. The first stage (FPA) improves the accuracy of query answers and is described using a trusted central sever. The second stage (DLPA) is used to obtain a distributed noise addition.

1) Fourier Perturbation Algorithm (FPA_k): The FPA_k algorithm answers a given query sequence Q of length n with small error under differential privacy. FPA_k is based on compressing the answers, Q(I), of the query sequence using an orthonormal transformation. This means finding a k-length compressed query sequence, F^k $= F_1^k$,……., F_k^k, where $k<<n$, such that the answers, F^k (I), can be used to approximately compute Q(I). Then F^k (I) is perturbed instead of Q(I) using a lower noise (the noise actually reduces by a factor of n/k) while preserving differential privacy. The error that creeps because Q(I) cannot be constructed exactly from F^k (I), is significantly lower than the perturbation error caused by adding noise directly to Q(I). PASTE uses Discrete Fourier Transform (DFT) to find F^k, the DFT compressed query sequence which has large L_1 sensitivity.
2) Distributed LPA (DLPA): To answer an aggregate-sum query sequence Q distributedly under differential privacy, DLPA - distributed version of the LPA algorithm is used. Our complete solution comprises of using FPA_k .

In case of single aggregate-sum query Q: the generalization to the sequence Q requires n separate invocations, once for each query Q_i in Q. Since Q is an aggregate-sum query, $Q(I)= \sum_{u=1}^{U} f_u(I_u)$ where the function f_u maps user u's data to numbers. $x_u=f_u(I_u)$, so that $Q(I)=\sum_{u=1}^{U} x_u$

To perturb Q(I), each user u adds a share of noise, n_u, to his private value x_u. The noise shares are chosen such that the estimation error is small and $\sum_{u=1}^{U} n_u$ is sufficient for differential privacy (but n_u alone is not sufficient so $x_u + n_u$ cannot directly be sent to the aggregator). The user u computes encryptions of $x_u + n_u$ and sends it to the aggregator. Due to encryption, the aggregator cannot learn anything about x_u. However, using the homomorphic property, it can compute the encryption of the sum of all received values from all users thereby getting an encryption E of Q' = $\sum_u x_u + n_u$. This encryption is then sent back to all users who use the threshold property to compute their respective decryption shares. Finally, the decryption shares are sent to the aggregator who combines them to compute the final decryption. The end result is a noisy differentially private estimate Q of Q(I).

To answer a query sequence Q of n aggregate-sum queries, the aggregator first uses FPA_k to compute k-length sequence F^k . Now, to perturb the answers of the query sequence F^k DLPA algorithm is applied on it. The aggregator gets a noisy differential estimate of $F^k(I)$. Then the aggregator computes that inverse DFT to reconstruct Q'. The final estimate Q' has error characteristics of FPA_k algorithm computed in distributed way from DLPA.

2.3.Practical differential privacy via grouping and smoothing

Problem Description - The third paper [3] by Kellaris et al in 2013, deals with the problem of one time publishing of non-overlapping counts with Є-differential privacy. For example, in a database where column j refers to a "check-in" place and record i corresponds to a user, we query to "return the number of users that checked in at each place". When we try to publish these counts for Є -differential privacy using technique given in [1], we realize that the published data of LPA might be meaningless because the applications that are being considered here have arbitrary sensitivity. For example, a user may check at numerous places and affect many counts.

Data set - The authors experimented with 4 real world data sets:

1. Checkin- This data set contains user and their corresponding check-in information from the Gowalla geo-social network
2. Document- This data set contains documents/abstracts and their vocabulary terms information from the Pub-med data set
3. Netflix- This is a training data set used in Netflix Prize competition. It contains information about the rating of a movie given by a particular user
4. Trajectory- This data set contains GPS traces collected by GeoPKDD project, mapped on a road network. It contains information about the trajectories and their corresponding network notes.

Methodology - They introduce a scheme called Grouping and Smoothing (GS) in which they achieve Є -differential privacy with high utility. The algorithm decomposes the database columns into disjoint groups(Grouping) and averages (Smoothing) the counts in each group. After that it adds Laplace noise to each group average. This diminishes the sensitivity and decreases the

Laplace noise scale. We also observe that in comparison with [2], GS approximates the count vector much better than FPA. The authors use real data sets to perform experiments which confirm the superiority of GS over LPA [1] and Є FPA [2].

The authors have considered a case with Database D, with n rows corresponding to users and d columns corresponding to attributes (e.g. places). The goal is to develop an effective grouping strategy for the database columns with low privacy cost. They have used sampling techniques to construct 2 types of Grouping and Smoothing (GS) schemes: GS-R and GS-S. They also augment GS-S with a fine-tuning step which determines an optimal group size while maintaining Є-Differential Privacy.

1. Random Grouping (GS-R)

Input to GS-R is database D with columns d, query Q, and its sensitivity $\Delta_1(Q)$. As shown in the figure, it computes a grouping strategy g. It also computes the answer vector c =Q(D). Next, it generates the vector c_g holding the average of the counts of columns and adds Laplace Noise = Lap(1/Є) to each of its elements. Eventually it constructs t, which stores the final output.

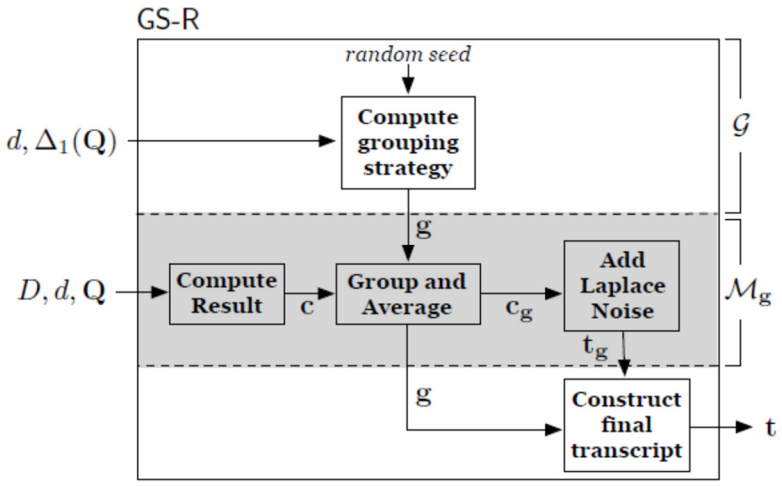


Figure 1: Outline of GS-R

2. Grouping via Sampling (GS-S)

Since GS-R randomly selects the groups, "bad" grouping strategies can occur. GS-S creates an optimal sample D_s of D, instead of D. The resulting c_s has much less noise than c in GS-R. GS-S has two differences compared to GS-R (i) the grouping strategy g is different (ii) after computing c_g based on c and g, it injects noise Lap(2/Є) to derive t_g instead of Lap(1/Є) in GS-R

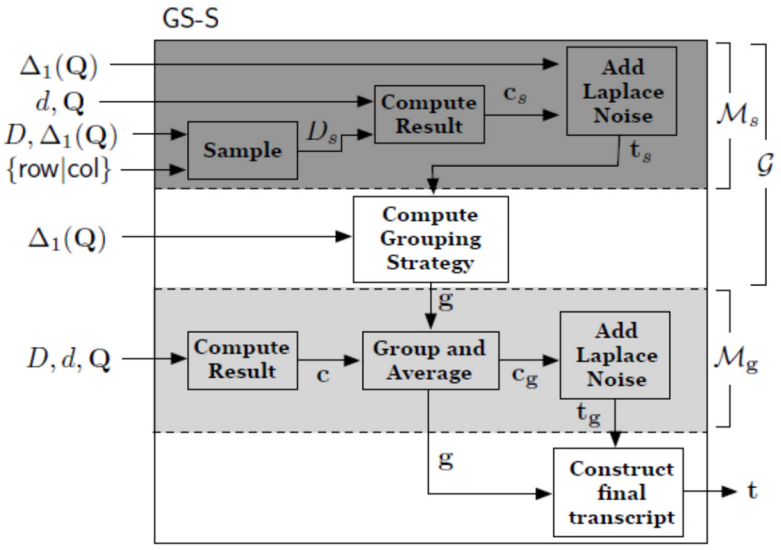


Figure 2: Outline of GS-S

2.4.A Practical Framework for Privacy-Preserving Data Analytics

Problem Description -The fourth paper [4] by Fan et al in 2015, deals with the case where each user could contribute large number of records, example, purchaser or check-in history of a user. The problem being considered here is two types of analytical tasks, one, top k-discovery (example query - Most popular places in a city X) and two, context aware recommendation (example query – good places to visit on Tuesday). They consider the problem of releasing a set of count queries to a database D (containing "n" users and m items of interest) related to the attributes of interest and customizable predicates to answer analytical questions. These are basically 2 problems of finding Item Counts (select * from D where *v1D=Vi*) and Edge Counts (select * from D where *v1D=Vi* and attrA=Aj).

We observe that to generate a database which satisfies Є-differential privacy with method used in [1], the more data a user contributes to an analysis process, more perturbation noise is introduced to hide his/her presence. This perturbation error is a cost and depends on the highest possible data contribution.

Datasets - The authors used four real-world data sets for the empirical study:
1. Gowalla – This is a data of location check in records collected among users based in Austin from Gowalla location-based social network by Berjani and Strufe [3] between June and October 2010.
2. Foursquare - This is also data of location check in records collected from Foursquare by Long et al. [18] between February and July 2012.

Record contains a user location and check in time stamp. The check in data is representative of class of services which value the returning behavior, such as buying or browsing.
3. Netflix - This data set consists of movie ratings and is the training data set for the Netflix Prize competition.
4. Movie-Lens - This is also a data set of movie ratings and is collected from users of Movie-Lens website. Each rating has a user, movie, rating score, time-stamp and other demographic information of users.

Methodology - In [4], the authors present two methods with different sampling techniques to draw a subset of the data for analysis. The first sampling technique is SRA, which randomly samples up to l records per user. The second sampling technique is HPA which selects up to l records from each user that are most useful for the specific analytical tasks. These techniques do not require input data to be preprocessed such as removing users with little or large data. And their experiment results show that the techniques outperform the prior approaches for performing K-top discovery and context aware recommendations.

1. Simple Random Algorithm (SRA): Maximum number of records contributed by each user (M) could be very large or even unbounded in real applications for e.g. a user could repeated check-in at the same location. As a result, very large perturbation noise (Laplace noise) is required to provide differential privacy. In this solution, the authors provoked the sample, the raw input dataset (D) and allow up to "l" records per user in the sample database (thus bounding the individual contribution to the database by a fixed constant. In the figure below, the SRA generates a sample database (D_R) by random sampling at most "l" records per user in D. Subsequently it computes the query answers to q_1 and q_2 from the sample database D_R and adds perturbation noise from Lap(l/€) to the result of each query.
   We also understand that the more we sample for each user the closer q_1 (D_R) and q_2 (D_R) are to the true results q_1 (D) and q_2 (D) respectively. There is a trade-off between accuracy and privacy for query q_1 to get an optimal choice of l.

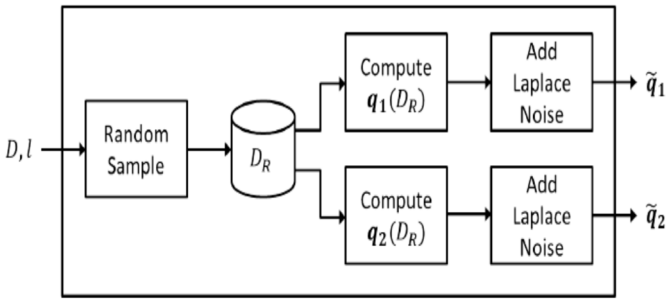


Figure 3: Outline of SRA Algorithm

2. Hand-Picked Algorithm (HPA): A lot of data analytical tasks try to find popular places or products such as in recommendation services. This means that some records of a particular user might be more useful for a data analytics purpose. The outline of HPA is the following two steps:
   a. Private estimation of record utility: The authors introduced the concept of "usefulness of a record" and define popularity based utility score for each record and propose to preserve records with highest scores for each user. The utility is also conducted on a sampled database D with the sampling factor d. D_E is obtained by randomly choosing up to d records per user from the raw database D.

   b. Greedy Sampling Procedure: Further, they propose to greedily pick up to l records with highest utility scores for each user's data in D. After the greedy sampling step, the results to q_1 and q_2 will be computed on the sampled database D_G. Each individual item count and edge count will be perturbed by Laplace noise from Lap ($1/€_1$) and Lap ($1/€_2$), respectively.

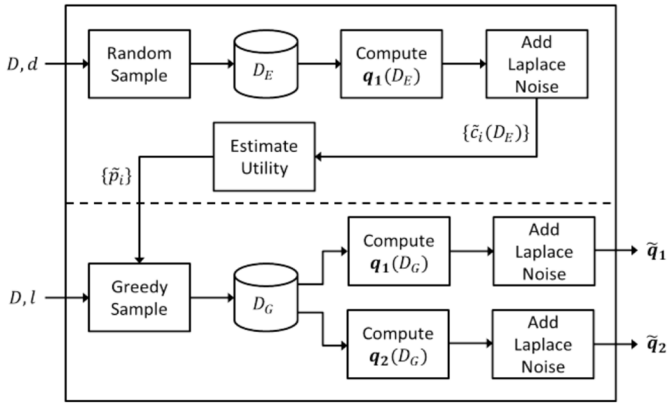


Figure 4: Outline of HPA Algorithm

2.5.Differentially Private Publishing of High-dimensional Data Using Sensitivity Control

Problem description - In the fifth paper by Day et al [5], the authors consider the problem of publishing column counts for high dimensional binary datasets while satisfying differential privacy. This problem is an example of generating meaningful query results from a high dimensional data without compromising on privacy of user data.

Datasets used – The authors consider a binary database made of 0,1 with number of columns as d. Count of columns is the number of 1's in a column. They argue this database is representative of databases such as transaction datasets where each title represents items purchased in a transaction or search datasets where each triple is keywords search by a user. Published column counts can represent important info such as popularity of items in online store or too used keywords in query log. For experimentation on approach paper uses 6 different datasets namely Netflix, Transaction, Movie lens, Document, AOL, Kosarak.

In most datasets number of columns is much larger than average column count, so the authors says that the approach of adding a tuple with all 1's as a noise, would overwhelm the data as sensitivity would become equal to number of columns (d) which is equal to number of extra 1's added.

To overcome this problem an upper limit Θ is set on amount of noise added, and if any rows are above this level they are normalized. So, issue becomes if to choose a threshold which is optimal since increasing Θ reduces errors by normalization but increases total errors due to noise. So far literature assumed Θ is set manually but paper proposes 2 systematic approaches DPsense and DPsense-S

Methodology -

DPsense- in this approach paper introduces a low sensitivity quality function that enables use of exponential mechanism to choose a good threshold Θ for limiting the sensitivity. Using this Θ threshold rows which are above this threshold is normalized through below formula

$$D|_\theta(i,j) = \begin{cases} D(i,j) & \text{if } RC_i \leq \theta \\ D(i,j)\frac{\theta}{RC_i} & \text{if } RC_i > \theta \end{cases}$$

Where $RC_i = \sum_{j=1}^{d} D(i,j)$ is the row count (sum of all cells in row i). This normalization causes truncation error while the noise errors are caused by Laplacian errors. Optimal Θ must minimize both these errors and paper achieves that by creating a quality function of both these errors

$$ac(D) = \frac{1}{d}\sum_{i=1}^{n}\sum_{j=1}^{d} D(i,j) = \frac{1}{d}\sum_{j=1}^{d} c_j$$

Where ac(D) denotes average column count in D and c_j is jth column count in database D. Below describes the quality function proposed by the paper using the intuition higher the average column count, lower the truncation error

$$q(D,\theta,\epsilon_p) = ac(D|_\theta) - \frac{\theta}{\epsilon_p}$$

DPsense-S - DPsense adds noise to column counts of normalized dataset and since normalized datasets column counts are always underestimated, DPsense has systematic under estimation bias. Paper proposes scaling approach using factor α to correct that and calls algorithm DPsense-Scaling. A magnifies noise and using this noise magnification and estimated error after scaling by Θ and α, paper estimates optional α and Θ. Average column count error with scaling is below

$$\begin{aligned} ae(D, D|_\theta) &= \frac{1}{d}\sum_{j=1}^{d}\left(\left|\alpha \cdot \sum_{i=1}^{n} D|_\theta(i,j) - \sum_{i=1}^{n} D(i,j)\right|\right) \\ &= \frac{1}{d}\sum_{j=1}^{d} |\alpha \cdot c_j^\theta - c_j| \end{aligned}$$

Where $|\alpha \cdot c_j^{\theta} - c_j|$ is the error after correcting the count in column j by scaling factor α. Below represents the quality function for DPSense-S algorithm.

$$qs(D, \theta, \alpha, \epsilon_p) = -ae(D, D|_\theta) - \alpha \cdot \frac{\theta}{\epsilon_p}$$

This quality function represents the correlation between α and Θ and the effect of their combination in minimizing the error.

Results - Algorithms of approaches ([1], [2], [3]) mentioned before were run 30 times and average was computed using 2 differential privacy budgets.

The research paper shows how the quality functions and Mean Absolute error(MAE) are symmetrical around horizontal line. Maximizing quality function minimizes MAE.

They also report that DPsense and DPsense-S Algorithms are significant improvements over other approaches for both MAE and MRE (mean relative error) and DPsense-S boosts performance over DPsense for Netflix, movielens, AOL and Kosark while under performs for transactions and document. These 2 datasets have some columns with very high counts contributed by rows with small total sums, and scaling factor over corrects these rows. The research points out that since it uses overall MAE and not just MAE for top columns, that causes this issue. Paper further evaluates that in case of top-k columns DPsense outperforms DPsense-S.

They state that further studies may evaluate this quality function approach on other problems of high dimensional databases.

3. Comparison

Below is the comparison of SRA, HPA, LPA, DFT and GS methods on the data sets of [4]. The utility of item counts and edge counts released by them are evaluated with Mean Squared Error (MSE), KL divergence, Top-K discovery.

1. MSE: This provides a generic utility comparison of different methods on released counts.
   - The LPA yields highest error in both item counts and edge counts.
   - The GS method shows results similar to DFT, except for Movie Lens item counts.
   - SRA and HPA have lowest MSE error except in Netflix item counts and Movie-Lens item/edge counts.

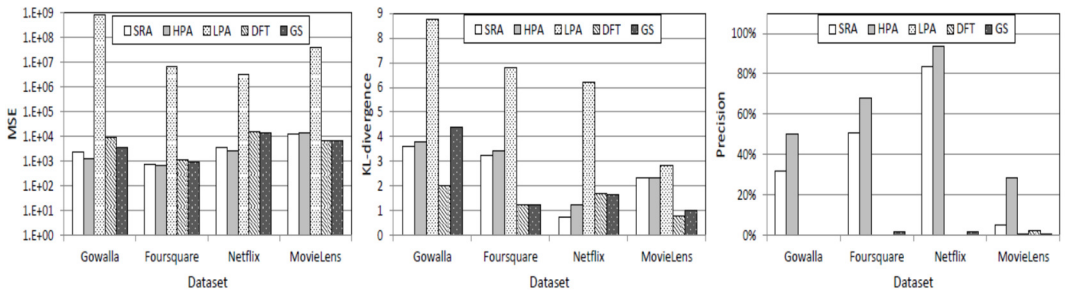
2. KL-divergence: This provides a measure of distance between the released distribution and original distribution for each query.
   - The released distribution by LPA is much farther from original data distribution from all other methods
   - DFT and GS mostly preserved the count distribution.

- SRA and HPA provide comparable performance to all other methods though they are not optimized to preserve distributional similarities.
- SRA beats HPA in approximating the two distribution.

3. Top-K Discovery: In this measure, the quality of the top 10 discovery by all the approaches discussed before is retrieved and evaluated. In the utility of released items count graph, we observe that no method except SRA and HPA preserves the ten most popular items in any data set. This is because LPA suffers from high perturbation error and DFT and GS yield over smooth release counts so we cannot differentiate the most popular items from those ranked next to them. HPA even achieves 100 per cent precision for Netflix data.

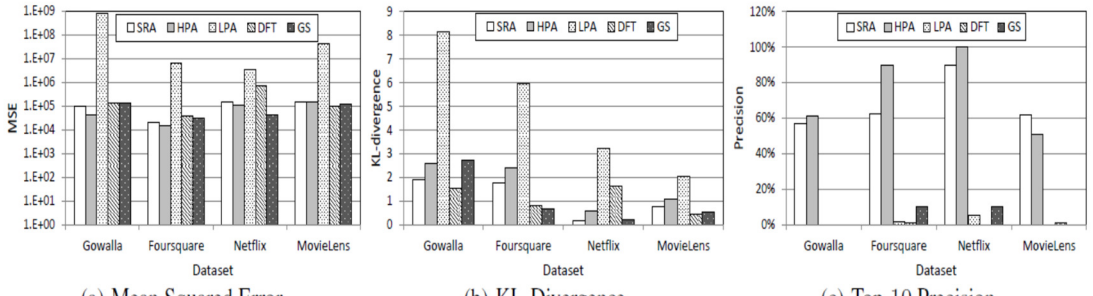
In the utility of released edge counts graph, we observed that mostly HPA outperforms SRA.

In the graph of top-K precision of the released item counts (k ranging from 1-1000) we see that the performance of HPA approach is 100% when k=1 and drops as *k* increases. The precision of SRA approach also decreases as *k* increases but the rate of decrease is less than HPA. LPA, DFT and GS show 0 percent precision when k=1 and higher precision as k increases. The SRA and HPA approaches work better compared to other approaches in discovering the most popular items up to k=100 but they do not discover less popular items due to lack of information in sample database.



(a) Mean Squared Error (b) KL-Divergence (c) Average Top-10 Precision

Figure 5: Utility of Released Item Counts



(a) Mean Squared Error (b) KL-Divergence (c) Top-10 Precision

Figure 6: Utility of Edge Item Counts

A comparison of approaches presented in [1], [2], [3] and [5] is given in research paper [5], however I have not included it here due to space constraints

4. Conclusion

The study in these papers successfully presented new and different ways for performing data analytics with Є-Differential Privacy. in different query scenarios involving big data. The research provides methods. The experimental evaluation results of these works on the real-world datasets confirm that they perform accurate data analytics and also prove their practicality in wide set of applications. Future work involves design of a system to implement these approaches in all real-time data applications. Also, we need to implement them for more complex data analytic tasks which involve multiple, overlapping count queries and statistical queries.

5. References

1. C. Dwork, F. McSherry, K. Nissim, and A. Smith. Calibrating Noise to Sensitivity in Private Data Analysis, In Proceedings of the 3rd Theory of Cryptography Conference, pages 265-284, Heidelberg, 2006. Springer-Verlag
2. V. Rastogi and S. Nath. Differentially private aggregation of distributed time-series with transformation and encryption, In proceedings of the 2010 ACM SIGMOD International Conference on Management of data, pages 735-746, 2010.
3. G. Kellaris and S. Papadopoulos. Practical differential privacy via grouping and smoothing, In proceedings of the 39th international conference on Very Large Data Bases, PVLDB'13, pages 301-312, 2013.
4. L. Fan, H. Jin. A Practical Framework for Privacy-Preserving Data Analytics, In Proceedings of International World Wide Web Conference Committee, IW3C2, 2015.
5. W. Day, N. Li. Differentially Private Publishing of High-Dimensional Data Using Sensitivity Control, In Proceedings of ASIA CCS, 2015