Explode: An Extensible Platform for Differentially Private Data Analysis

Emir Esmerdag*, Mehmet Emre Gursoy†, Ali Inan‡ and Yucel Saygin§
*Istanbul Technical University. Istanbul, Turkey. Email: emiresmerdag@gmail.com
†Georgia Institute of Technology. Atlanta, GA. Email: memregursoy@gatech.edu
‡Adana Science and Technology University. Adana, Turkey. Email: ainan@adanabtu.edu.tr
§Sabanci University. Istanbul, Turkey. Email: ysaygin@sabanciuniv.edu

***Abstract*—Differential privacy (DP) has emerged as a popular standard for privacy protection and received great attention from the research community. However, practitioners often find DP cumbersome to implement, since it requires additional protocols (e.g., for randomized response, noise addition) and changes to existing database systems. To avoid these issues we introduce Explode, a platform for differentially private data analysis. The power of Explode comes from its ease of deployment and use: The data owner can install Explode on top of an SQL server, without modifying any existing components. Explode then hosts a web application that allows users to conveniently perform many popular data analysis tasks through a graphical user interface, e.g., issuing statistical queries, classification, correlation analysis. Explode automatically converts these tasks to collections of SQL queries, and uses the techniques in [3] to determine the right amount of noise that should be added to satisfy DP while producing high utility outputs. This paper describes the current implementation of Explode, together with potential improvements and extensions.**

***Keywords*—Differential privacy; privacy protection; data mining; relational databases**

I. Introduction

Vast amounts of data can be collected and stored via modern computer systems. Although mining this data has clear benefits, privacy remains a major obstacle. Differential privacy (DP) is currently considered the state of the art in privacy protection. Yet, practical deployment of DP has not been successful due to practitioners' and data owners' concerns. Among leading concerns are that: (i) DP is difficult to understand, and users need to be experts in DP to use a DP system. (ii) DP is not compatible with existing systems, and thus implementing it is a burden.

To address these challenges, we propose Explode, a differentially private data analysis platform. It appeals to the data owner because it works on top of SQL databases (still a widely deployed database standard), is easy to install and works with no changes to existing infrastructure. It appeals to the non-expert user because its computation and algorithms are transparent to the user. The user need not have any background in DP, and only interacts with Explode through a graphical web interface. Explode is designed to feel like a high-level automated data analysis platform.

The development of Explode was fueled by the theoretical findings in [3]. DP ensures privacy by adding noise to the outputs of algorithms or queries. A major difficulty in DP is that computing the noise magnitude for DP is NP-hard, since calculating the *sensitivity* of an algorithm or query set is NP-hard [9]. Therefore, research in DP has focused on writing or modifying an algorithm so that it has a fixed, well-defined sensitivity that can be calculated easily. This is in conflict with the need to implement arbitrary data analysis tasks for which sensitivity is unknown or unclear in advance. This is where the theoretical contribution of Explode lies: We efficiently find a (tight) upper bound on sensitivity using the techniques presented in [3], which allows Explode to add noise to satisfy DP in arbitrary tasks - as long as the data accesses of the task can be expressed as a serial or parallel execution of statistical queries.

We believe that Explode is beneficial both to researchers and practitioners. From a researcher's point of view, Explode is an extensible platform on which many DP data analysis tasks can be implemented, their accuracy can be compared and new datasets can be privately studied. For example, upon devising a new algorithm, the researcher can implement it on Explode and compare it to existing algorithms to see if the new algorithm performs better. In addition, a researcher may run one algorithm with different privacy levels and parameters, to observe how the outcome of the algorithm changes according to the parameters. From a practitioner's point of view, Explode can be used as a simple add-on to protect their private data with DP, while providing users with a platform with various data analysis capabilities.

Comparison to Related Work. We report the main differences between Explode and the other works enabling practical differentially private data analysis, with the purpose of better clarifying where our work stands. In [5], McSherry introduces PINQ, a platform that facilitates users' development of differentially private programs in the LINQ language. PINQ provides operators for queries that have fixed sensitivity, and does not perform sensitivity analysis for complex programs. Also, while users have to write their own programs in PINQ, Explode comes with several ready-to-use programs. In [8], Roy et al. introduce Airavat, a privacy-preserving framework for MapReduce computation on the cloud. Airavat aims to bound the information leakage in cloud computation, which is fundamentally different than our setting (client-server) and infrastructure. In [6], GUPT is proposed. GUPT's main contribution is that it allocates a privacy budget to each task

based on the expected accuracy of that task's output. As such, it eliminates clients' budget management. In [7], Reed and Pierce propose Fuzz, a functional programming language with a calculus that supports the generation of differentially private functions. For functions written in Fuzz, sensitivity is always well-defined and bounded. The use of Fuzz requires the data owner to get acquainted with the language, while Explode has no such requirement. Simultaneous to our work, Hay et al. developed DPComp [1]. DPComp is a web-based system in which users can assess state of the art DP algorithms on various datasets, and contribute new algorithms and datasets. DPComp only considers algorithms that answer 1D and 2D range-count queries, whereas we consider also more complex data analysis tasks.

II. System Overview

A. Preliminaries

Differential Privacy. Informally, an algorithm $\mathcal{A}$ is differentially private if its behavior is insensitive to a small change in the input database. Formally, $\mathcal{A}$ is ε-differentially private (ε-DP) if for all neighboring databases $\mathcal{D}, \mathcal{D}'$ and for all possible outcomes $S \subseteq \mathit{Range}(\mathcal{A})$, $Pr[\mathcal{A}(\mathcal{D}) \in S] \leq e^{\varepsilon} \times Pr[\mathcal{A}(\mathcal{D}') \in S]$. Two databases $\mathcal{D}, \mathcal{D}'$ are called neighboring databases, if one can be obtained from the other by changing only one tuple.

A prominent method in satisfying DP is by perturbing the output of algorithm $\mathcal{A}$ by random noise. The noise depends on the L_1 sensitivity of the algorithm, which measures how much the output of $\mathcal{A}$ can change due to a change in one data tuple. Upon measuring the sensitivity of $\mathcal{A}$, one can use the Laplace mechanism to achieve DP: Let S_{L_1} denote the L_1 sensitivity and $Lap(\lambda)$ denote a random variable sampled from the Laplace distribution with mean 0 and scale parameter λ. For a query $q : \mathcal{D} \rightarrow \mathbb{R}$, the algorithm $\mathcal{A}$ that answers q by $\mathcal{A}(q, \mathcal{D}) = q(\mathcal{D}) + Lap(\lambda)$ is ε-DP if $\lambda \geq S_{L_1}/\varepsilon$.

λ is often called the noise magnitude, and directly depends on S_{L_1}. However, computing S_{L_1} for arbitrary query sets or data analysis tasks is NP-hard. As mentioned earlier, our theoretical contribution lies in this area: Since we can calculate an upper bound on S_{L_1} for any query set, as long as the algorithm $\mathcal{A}$ can be expressed as a serial or parallel execution of several query sets, its ε-DP implementation using Explode is easy.

Data Model. We assume that data is stored in a central SQL server. SQL databases are still among the most popular databases used in practice. We allow databases with multiple tables, as long as the tables are not connected, and algorithms and queries are executed on only one table at a time. This is because a `JOIN` operation has potentially unbounded sensitivity [5].

Each table in the database may contain several attributes (i.e., columns). We make the following assumptions regarding the attributes: (i) The domain of each attribute is finite. Finite domains allow bounding the effect of a single record on the output of domain-specific aggregate functions, such as `SUM`. (ii) Attributes are either numeric, categorical or ordinal. Some attribute types (e.g., binary objects, dates) can be easily transformed into numeric values. Other attribute types (e.g., strings) cannot be supported, due to the difficulty in converting their domain into a finite, well-defined set of values.

Query Model. Differential privacy allows only statistical queries. Further, Explode works with statistical *range* queries written in SQL (since we assume SQL as our underlying database). We say that such queries follow the following form:

```
SELECT AGG
FROM T
WHERE pred(A_1) AND ... AND pred(A_d)
```

where AGG is an aggregate function (e.g., `COUNT`, `SUM`, `MIN`, `MAX`) and `pred(`A_i`)` is a predicate on attribute A_i of table T. Example predicates are $A_i \geq x$, A_i BETWEEN x AND y etc. Disjunctive predicates (i.e., predicates involving x OR y) are not allowed.

In essence, queries in this model build d-dimensional hyperrectangles as their range, and retrieve a statistical property (e.g., `COUNT` or `MIN`) of that hyperrectangle. We require that the data accesses of data analysis tasks conform to this grammar. Once queries are issued and (noisy) results are retrieved, the data analysis task may use the results in any (however complex) way it desires, e.g., calculate conditional probabilities using `COUNT`s, find entropy between attributes etc.

B. System Design

Implementation Details. Explode was developed primarily in `node.js`, and runs on 64-bit Linux systems. During installation, it requires connection parameters to a SQL server. We tested Explode on 64-bit Ubuntu 14.04, `node.js` version 4.4.3 and MySQL version 5.5.

Client-Server Setting. Once Explode is successfully installed, it acts as a server that hosts ε-DP data analysis tasks for its clients. The clients need to sign up to use Explode. We use the standard username-password combination to authenticate clients. Authentication is handled by `node.js`'s Passport[1] middleware.

Budget Management. Each user has a non-negative privacy budget for each table in the database. While running a task X on table T, the user specifies how much budget s/he would like to spend on X. Upon the successful completion of X, the specified amount is deducted from the user's budget for T. The user's total expenditure cannot exceed their initial budget.

For example, let Alice start with budget $\varepsilon_A = 2$ for the *Census* table. Alice spends budget $\varepsilon_1 = 1$ for the first task, $\varepsilon_2 = 0.3$ for the second task and $\varepsilon_3 = 0.7$ for the third task. By the sequential composition property of DP, her access to *Census* data is ε_A differentially private, since $\varepsilon_A = 2 \geq \varepsilon_1 + \varepsilon_2 + \varepsilon_3$. Now, if Alice would like to run a fourth task with ε_4, since the total of her εs exceeds her initial budget ε_A she will not be permitted. Similarly, if Alice had tried to set $\varepsilon_3 = 0.8$ for her third task, she would not have been permitted.

[1] http://passportjs.org/

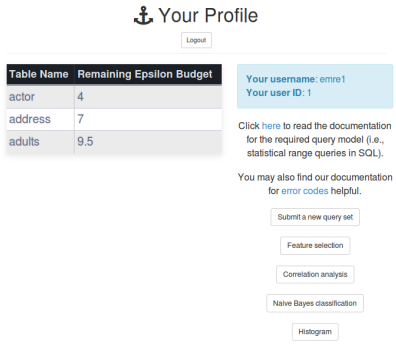

Fig. 1. Profile page

Users' profile page (i.e., the page they land on after logging in) summarizes their remaining budgets. An excerpt is provided in Fig. 1. Also, for easier management of users' budgets, we supply the data owner with simple scripts that update (e.g., reset, delete) budgets.

Authentication & Security. Since users have privacy budgets (which are essentially treated as their currency); it is critical to protect their accounts and the data they retrieve from the server. For example, if Alice can view or hijack responses to Bob's queries, this defeats the purpose of having separate budgets for Alice and Bob, since Alice can learn private information without spending any of her budget.

Explode manages users' security via creating encrypted sessions between the server and its clients. We use standard SSL for this purpose. On the client's side, session-related information is managed by a cookie. On the server's side, Explode creates a job scheduler for maintenance. The job scheduler runs periodically to check for inactive sessions and ends them.

C. Current Features

In this section we list the current capabilities and data analysis tasks supported by Explode.

Statistical Query Sets for Data Analysis. We support answering arbitrary statistical query sets written in SQL. These are subject to the data and query models given in Section II-A, e.g., if the user poses a non-statistical query trying to fetch actual rows from the database, then the query will not be answered.

To use this functionality, the user types or uploads his query set and specifies the privacy budget he would like to spend. Using our sensitivity calculation strategies, a bound on the noise magnitude is obtained, and answers are returned using the Laplace mechanism.

Feature Selection. Feature selection is the process of choosing a subset of relevant features (e.g., predictor attributes) before constructing a model. Explode performs feature selection by privately calculating the *entropy* between attributes. Given a table, a *class* attribute, ε and k, it retrieves the top-k attributes with lowest entropy to the *class* attribute. These columns are the most homogeneous with *class*, and are therefore good indicators of the *class* values.

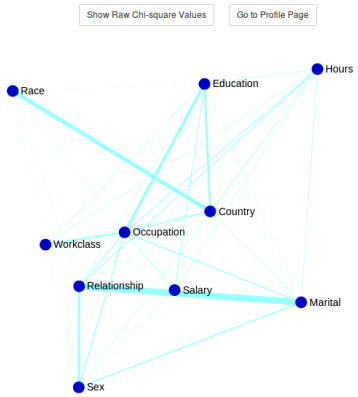

Fig. 2. Visualizing correlation between attributes in the Adult dataset

Correlation Analysis. Another interesting task is to analyze the correlation between the attributes of a table. For this purpose, Explode uses χ^2 correlation tests. Consider calculating Pearson's χ^2 test statistic for two categorical attributes $X = \{x_1, ..., x_n\}$ and $Y = \{y_1, ..., y_m\}$:

$$\chi^2 = \sum_i \sum_j \frac{(O_{ij} - E_{ij})^2}{E_{ij}}$$

The "observed" values, O_{ij}, are retrieved via `COUNT` queries: `SELECT COUNT(*) FROM T WHERE` $X = x_i$ `AND` $Y = y_j$. The "expected" values, $E_{ij} = a \times b/t$, where a is the answer to `SELECT COUNT(*) FROM T WHERE` $X = x_i$, b is the answer to `SELECT COUNT(*) FROM T WHERE` $Y = y_j$, and t is the answer to `SELECT COUNT(*) FROM T`. t is considered public knowledge for the given definition of DP, and hence we could do without adding noise to t. The query set for feature selection constitutes of queries in the above form. Answering this query set via DP is the same as answering any arbitrary query set.

We performed the χ^2 test for the *Adult* dataset retrieved from the UCI Machine Learning Repository[2]. Using Explode, we found the levels of correlation between attributes and visualized them, as in Fig. 2. Note that even though the correlation analysis was ε-DP, the correlation levels are consistent with what one would expect without noise. For example, there appears to be a strong correlation between *Relationship* and *Marital* status, *Race* and *Country*, and *Education* and *Occupation*. However, there is a weak correlation between *Race* and *Education*, and *Hours Worked* and *Salary*.

Classification. Explode builds Naive Bayes Classifiers (NBC) to classify instances. The training data sits in the data owner's database, and is private. The test data is supplied by the user. That is, the user has some tuples that he would like to classify, but the classification process needs to be ε-DP.

Given a table T, class attribute C with domain $\Omega(C)$, n predictor attributes (an appropriate set of predictors can be chosen via, e.g., feature selection) and their values $x_1, ..., x_n$

[2]http://archive.ics.uci.edu/ml/datasets/Adult

for an instance to be classified, NBC assigns the following class label:

$$\underset{C_k \in \Omega(C)}{\arg\max} \; [p(C_k) \times \prod_{i=1}^{n} p(x_i|C_k)]$$

In the above, $p(x_i|C_k)$ is simply:

$$p(x_i|C_k) = \frac{\text{Count instances w/ } x_i \text{ and } C_k}{\text{Count instances w/ } C_k}$$

and $p(C_k)$ is:

$$p(C_k) = \frac{\text{Count instances w/ } C_k}{\text{Count all instances}}$$

Counting instances translate to issuing `COUNT` queries on the private database. Thus, the query set for NBC consists of queries that calculate the probabilities above. After obtaining (noisy) answers to these queries, an NBC model can be built, and instances can be classified.

Histogram Publishing. Interestingly, in the last few years a great focus was placed on accurately answering histogram and `COUNT` queries while preserving DP. Many algorithms were implemented solely for this purpose. We identified a state of the art algorithm [10] (benchmarked against other algorithms in [2]) and implemented this algorithm in Explode, by expressing its data accesses as statistical SQL queries. Our goals by implementing [10] are: (i) For a fixed privacy budget, by using a state of the art algorithm we produce histograms that are more accurate than previous algorithms. (ii) We show that works/algorithms independent from Explode can be added to Explode if desired. To make histogram results more user-friendly, in addition to yielding the private answers in plain text format, we also visualize the histograms.

Adding New Features. Any data analysis task whose data accesses can be expressed as a collection of statistical SQL queries, and relies on the Laplace and exponential mechanisms [4] of DP can be added to Explode. As we continue to develop Explode, we plan to add new tasks (e.g., clustering) to make Explode a more complete data analysis platform.

On the other hand, Explode also has some limitations. These can be discussed in two categories: (i) Limitations due to DP: Since we employ DP as our privacy protection standard, we need to place requirements on our data and query models. These were given in Section II-A. (ii) Limitations due to SQL: Not all algorithms can be expressed as a collection of statistical SQL queries. In addition, Explode's performance inherently depends on the number of SQL queries a task will require (for complex tasks, this can be many) and the SQL server's efficiency in answering them.

III. Demonstration Description

A. Demonstration Plan

Explode is ready to demonstrate. All components and features described in this paper have been implemented. For implementation details, we refer the reader to Section II-B. We will use commercial hardware to host a copy of Explode at the demo session.

B. Interaction and User Experience

We will prepare a local server and database containing popular benchmark datasets (e.g., the Adult dataset). We will encourage participants (i.e., users) to directly interact with Explode by creating an account or using an existing account. The participants will therefore be able to test all features and design components of Explode. We will also prepare sample inputs (e.g., sample classification instances) in order to quickly demonstrate some features of our system.

Explode has an easy-to-use graphical user interface, and its algorithms are transparent to the user. It is designed to feel like a high-level data analysis suite. Thus, participants need not be acquainted with DP or any of the theoretical aspects of our algorithms.

IV. Conclusion

We propose to demonstrate Explode, a differentially-private data analysis platform. Explode provides a convenient service for clients to analyze private data, while protecting the privacy of the data. Explode is powered by a sensitivity calculation technique that determines the required amount of noise for DP for an arbitrary set of statistical queries. Thus, any data analysis task whose data accesses can be reduced to a set of statistical queries can be added to Explode. To demonstrate this, we added several proof-of-concept tasks, e.g., Naive Bayes classification, feature selection etc. In addition, we also added a state of the art algorithm for histogram publishing, to demonstrate the plausibility of implementing independent, recent research in DP. As we improve Explode, we plan to add new components and support for more data analysis tasks.

Acknowledgment

This research was funded by The Scientific and Technological Research Council of Turkey (TUBITAK) under grant number 114E261.

References

[1] Hay, M., Machanavajjhala, A., Miklau, G., Chen, Y., Zhang, D., & Bissias, G. (2016, June). Exploring Privacy-Accuracy Tradeoffs using DPComp. In *Proceedings of SIGMOD'16* (pp. 2101-2104). ACM.

[2] Hay, M., Machanavajjhala, A., Miklau, G., Chen, Y., & Zhang, D. (2015). Principled Evaluation of Differentially Private Algorithms using DPBench. *arXiv preprint arXiv:1512.04817*.

[3] Inan, A., Gursoy, M. E., Esmerdag, E., & Saygin, Y. (2016, July). Graph-based modelling of query sets for differential privacy. In *Proceedings of SSDBM'16* (p. 3). ACM.

[4] McSherry, F., & Talwar, K. (2007, October). Mechanism design via differential privacy. In *FOCS'07* (pp. 94-103). IEEE.

[5] McSherry, F. D. (2009, June). Privacy integrated queries: an extensible platform for privacy-preserving data analysis. In *Proceedings of SIGMOD'09* (pp. 19-30). ACM.

[6] Mohan, P., Thakurta, A., Shi, E., Song, D., & Culler, D. (2012, May). GUPT: privacy preserving data analysis made easy. In *Proceedings of SIGMOD'12* (pp. 349-360). ACM.

[7] Reed, J., & Pierce, B. C. (2010, September). Distance makes the types grow stronger: a calculus for differential privacy. In *ACM SIGPLAN Notices* (Vol. 45, No. 9, pp. 157-168). ACM.

[8] Roy, I., Setty, S. T., Kilzer, A., Shmatikov, V., & Witchel, E. (2010, April). Airavat: Security and Privacy for MapReduce. In *Proceedings of USENIX NSDI'10* (Vol. 10, pp. 297-312). USENIX Association.

[9] Xiao, X., & Tao, Y. (2008). Output perturbation with query relaxation. *Proceedings of VLDB'08*, 1(1), 857-869.

[10] Zhang, X., Chen, R., Xu, J., Meng, X., & Xie, Y. (2014). Towards Accurate Histogram Publication under Differential Privacy. In *SIAM International Conference on Data Mining* (pp. 587-595).