DIAS: Differentially Private Interactive Algorithm Selection using Pythia

Ios Kotsogiannis  
Duke University  
iosk@cs.duke.edu

Michael Hay  
Colgate University  
mhay@colgate.edu

Ashwin Machanavajjhala  
Duke University  
aswinn@cs.duke.edu

Gerome Miklau  
University of Massachusetts  
miklau@cs.umass.edu

Margaret Orr  
Colgate University  
morr@colgate.edu

ABSTRACT

Differential privacy has emerged as the dominant privacy standard for data analysis. Its wide acceptance has led to significant development of algorithms that meet this rigorous standard. For some tasks, such as the task of answering low dimensional counting queries, dozens of algorithms have been proposed. However, no single algorithm has emerged as the dominant performer, and in fact, algorithm performance varies drastically across inputs. Thus, it’s not clear how to select an algorithm for a particular task, and choosing the wrong algorithm might lead to significant degradation in terms of analysis accuracy. We believe that the difficulty of algorithm selection is one factor limiting the adoption of differential privacy in real systems. In this demonstration we present DIAS (Differentially-private Interactive Algorithm Selection), an educational privacy game. Users are asked to perform algorithm selection for a variety of inputs and compare the performance of their choices against that of Pythia, an automated algorithm selection framework. Our hope is that by the end of the game users will understand the importance of algorithm selection and most importantly will have a good grasp on how to use differentially private algorithms for their own applications.

1. INTRODUCTION

In the modern age of big data, not only is information about individuals being collected by various agencies (e.g., hospitals, retailers, etc.), but also users voluntarily share their own data. Performing analyses on such data is tremendously valuable both for commercial and research purposes. Unfortunately, such analyses can lead to significant privacy breaches. Differential privacy has emerged as a gold standard privacy definition. Informally, differential privacy requires that the output of an analysis algorithm not change too much with the addition or removal of any single individual from the input dataset.

The interest in differential privacy within the research community has lead to a rich literature of algorithms. Most differentially private algorithms work by carefully injecting a certain amount of structured noise into analysis computations. General purpose algorithms like the Laplace Mechanism [1] are easy to adapt for a variety of tasks, but often offer sub-optimal error rates. Because of this, more sophisticated task-specific algorithms have been designed that are capable of reducing error rates by an order of magnitude while satisfying the same privacy guarantee. Some of these algorithms achieve lower error by adapting the added noise to specific properties of the data. This makes their performance data-dependent, meaning their error rates vary by input and deriving tight bounds on the error for a specific input is non-trivial. Moreover, a recent empirical study of 16 differentially private algorithms found that (a) there is no single dominating algorithm across all inputs, (b) the error rates of a single algorithm vary vastly depending on properties of the input include the dataset size, the setting of the privacy parameter, and other structural properties [2].

As a result, the rich literature on differentially private algorithm design has limited accessibility for a practitioner. In the current algorithm landscape, a practitioner needs to know details of each particular algorithm and under what conditions it is likely to perform well. Moreover, the fact that there is no single algorithm that dominates only makes the problem more challenging.

For this reason, in our paper that appears in SIGMOD 2017 [3], we formalize the problem of Algorithm Selection under differential privacy and propose Pythia, an end-to-end differentially private solution to the algorithm selection problem. Our vision with Pythia is to make differential privacy more accessible to data curators regardless of their expertise. Pythia is the first meta-algorithm for answering low dimensional queries on datasets under differential privacy. From the user’s perspective, Pythia is no different than any other differentially private algorithm for the task as it shares a common interface with them. Pythia offers an end-to-end differentially private solution, highly competitive error rates, and an effortless application to new tasks (i.e., users of Pythia don’t need to be privacy experts). We believe that Pythia is a necessary step towards a future where the practitioner specifies her privacy constraints and the queries she would like answered on a sensitive input and the differentially private system computes an optimized output under the privacy constraints.
Demo Overview

In this demonstration, users are introduced to two distinct but closely related concepts: (a) the importance and hard-
ness of algorithm selection in the context of differential pri-
vacy and (b) the impact of input properties on the error of
each algorithm. To achieve both ends, users play a data re-
lease game called DIAS (Differentially private Interactive
Algorithm Selection). The goal of the game is to perform
a complex data analysis task under differential privacy with
the highest possible accuracy. The task requires the simul-
taneous private release of multiple histograms of the original
data. Each player in the game is presented with the chal-
lenge of algorithm selection: given a set of algorithms to
choose from, they must select what they believe is the best
algorithm for each histogram task, with the wrong choice
leading to potentially significant loss in accuracy.

The game is organized in rounds and in each successive
round the complexity of the inputs increases and the users
are exposed to increasingly more sophisticated challenges of
algorithm selection under differential privacy. These chal-
enges are centered around input properties and how they
affect different algorithms. For example, in earlier rounds
users are introduced to the importance of the histogram’s
domain size and they choose only from a small class of sim-
pler algorithms. In contrast, in later rounds of the game,
users have to choose from all available algorithms and the
histograms to be computed have different structural proper-
ties that make algorithm selection challenging. At the end of
the game, users compare themselves with the best baseline
task, with the wrong choice
leading to potentially significant loss in accuracy.

The audience for the game includes both SIGMOD at-
tendees with little background in differential privacy – each
round is effectively a mini-tutorial – as well as differential
privacy experts who can compete to outperform Pythia.

2. PRELIMINARIES

Data Model. A database \( D \) is a multiset of records, each
having \( k \) attributes with discrete and ordered domains. Let
\( \mathcal{D} \) denote the universe of all possible input databases.
Following convention, we describe \( D \) as a vector \( x \in \mathbb{N}^k \) where
\( x_i \) reports the number of records type \( i \) for all \( d \) possible
types where \( d = d_1 \times \ldots \times d_k \) and \( d_j \) is the domain size of
the \( j \)th attribute.

Queries. A query workload \( \mathbf{W} \) is a set of \( m \) linear count-
ing queries defined on \( x \). This class of queries includes
queries that count the number of individuals satisfying a
range predicate on one or more attributes, and thus includes
histograms, marginals, and datacubes, in addition to more
general predicate counting queries. The answer to this work-
load is denoted as \( \mathbf{y} = \mathbf{W} \mathbf{x} \).

Differential Privacy. Differential privacy is satisfied when
the output distribution of the algorithm changes by only
a small multiplicative factor with the addition or deletion
of single record. Let \( \text{nbrs}(D) \) denote the set of databases
differing from \( D \) in at most one record; i.e., if \( D' \in \text{nbrs}(D) \),
then \( |(D - D') \cup (D' - D)| = 1 \).

Definition 2.1 (Differential Privacy [1]). A randomized al-
gorithm \( A \) is \( \epsilon \)-differentially private if for any instance \( D \),
any \( D' \in \text{nbrs}(D) \), and any subset of outputs \( S \subseteq \text{Range}(A) \),
\[
\Pr[A(D) \in S] \leq \exp(\epsilon) \times \Pr[A(D') \in S]
\]
\( \epsilon \) is called the privacy budget as it (indirectly) constrains
the amount of utility that can be extracted from the input.

Algorithms. The algorithms considered here take as input
a triple \((\mathbf{W}, \mathbf{x}, \epsilon)\) corresponding to a workload \( \mathbf{W} \), a private
dataset \( \mathbf{x} \), and a specific setting of the privacy parameter \( \epsilon \)
and they compute noisy answers to the workload \( \mathbf{W} \) on \( \mathbf{x} \)
that satisfy \( \epsilon \)-Differential Privacy, denoted \( \mathbf{y} \).

Differentially private algorithms can be broadly classified
into two categories: data-independent and data-dependent.
A data independent algorithm has the property that its er-
ror rate is independent of the input database instance. Cla-
ssic algorithms like the Laplace mechanism [1] are data in-
dependent. For the task of answering range queries on a
single dimension, the Laplace mechanism has the least error
when the domain of the attribute is small, whereas other
data independent techniques like \( \Pi \), that perform hierar-
chical decompositions of the domain can yield significantly
lower error rates for attributes with larger domains.

In many settings, however, the best performing algorithms
are data dependent. Examples of such algorithms include
DAWA, AGrid, AHP, and MWEM. These algorithms typi-
cally adapt to the particular dataset, finding a collection of
aggregate statistics that serve as an accurate approximation
of the underlying database. For instance, a popular data
adaptive strategy (employed by DAWA, AGrid and AHP)
is to first learn a partitioning of \( \mathbf{x} \) for which the data distri-
bution within each partition is approximately uniform, and
then summarize the dataset at the coarser granularity of par-
titions. Hay et al. [2] offer a more comprehensive overview
of data dependent algorithms.

A challenge with using data dependent algorithms is that
their error rates depend on the input database instance and
thus their performance can be hard to predict \textit{a priori}. This
motivates the problem of algorithm selection.

Problem Statement. Given a collection of state-of-the-art
differentially private algorithms, the data curator must se-
lect the algorithm that is likely to yield the best performance
on the curator’s data. This problem is formalized as follows.

Definition 2.2. Algorithm Selection [3]. Let \( A \) denote
a set of differentially private algorithms. Given the triplet
\((\mathbf{W}, \mathbf{x}, \epsilon)\) corresponding to a workload, a private dataset and
a setting of the privacy parameter \( \epsilon \) respectively, the algo-
rum selection problem is to select an algorithm \( A^* \in A \) to
answer \( \mathbf{W} \) on \( \mathbf{x} \) such that \( \epsilon \)-differential privacy is satisfied.

Solutions to algorithm selection must satisfy the following
three desiderata:

1. Differential privacy: The algorithm selector must itself
be differentially private. In particular, any use of the in-
put data in selecting an algorithm must be included in an
end-to-end guarantee of privacy. The obvious approach
of running all available algorithms on the sensitive input,
checking their error, and selecting the one with least error
has been shown to violate privacy [3].

2. Agnostic: Each algorithm \( A \in A \) should be treated as a
black box, i.e., only requiring that the algorithm satisfy
differential privacy. Agnostic methods are easier to use
Pythia

Algorithm Selection

Feature Extractor

Feature-based
Algorithm Selector

Run Algorithm

Private Answers of
Queries on Sensitive DB

Figure 1: The Pythia meta-algorithm computes private query answers given the input data, workload, and epsilon. Internally, it maintains a model of the performance of a set of algorithms, automatically selects one, and executes it.

3. Competitive: It should select an algorithm $A^*$ that offers low error rates on a wide variety of inputs.

Performance. The performance of an algorithm selector is measured using regret, which compares the error of the selected algorithm to the error of the best possible algorithm for that particular input.

Definition 2.3 (Regret). Given a set of differentially private algorithms $A$ and triplet $(W, x, \epsilon)$, the regret of selected algorithm $A \in A$ is:

$$\text{regret}(A, W, x, \epsilon) = \frac{\text{error}(A, W, x, \epsilon)}{\text{OPT}_A(W, x, \epsilon)}$$

where $\text{OPT}_A(W, x, \epsilon) = \min_{A \in A} \text{error}(A, W, x, \epsilon)$ and $\text{error}(A, W, x, \epsilon) = \|y - y\|_2$.

3. PYTHIA

Pythia is a differentially private meta-algorithm that solves algorithm selection problem and satisfies the three desiderata outlined in the previous section. It works in three steps (see Fig. 1). First, it extracts a set of noisy features from the input $x$ using a small fraction of the privacy budget. Next, it consults a Feature-based Algorithm Selector (FAS) and chooses one out of a library of differentially private algorithms based on the extracted features. Finally, it executes the chosen algorithm on the input using the remainder of the privacy budget. Since some of the privacy budget is used for feature extraction, Pythia will necessarily have slightly higher error than the optimal choice algorithm.

Algorithm selection is facilitated by the FAS, which is implemented in Pythia as a decision tree. The FAS is learned using an offline algorithm called Delphi that uses a synthetically generated training dataset constructed from publicly available inputs. For a detailed description of both Delphi and Pythia we refer the reader to the forthcoming paper [3].

For non-experts and are also readily extensible as new algorithms can be easily added.

By design, Pythia satisfies the first two desiderata (differentially private and agnostic) and we empirically show that it is also highly competitive, offering near-optimal regret rates for a wide variety of inputs. Pythia closes the accessibility gap for differential privacy since it does not require from the practitioner any knowledge of differentially private algorithms. Lastly, Delphi’s design allows the fast and easy inclusion of new algorithms as they are proposed by the research community.

4. DEMO OVERVIEW

Target Group. The audience for DIAS will include SIGMOD attendees who have little prior knowledge of differentially private algorithms as well as experts in differential privacy. This demonstration caters to both groups, privacy experts who participate in DIAS compete against Pythia for the task of algorithm selection and can see how well they fair against our automated system. At the same time, non-experts who are interested in differential privacy and want to understand the subtle nuances of differentially private algorithms have a chance to do so by participating in DIAS, since its rounds are designed to serve as a brief tutorial.

Game Organization. DIAS is a game of algorithm selection, where users play by selecting algorithms for a variety of different histograms. Once users have selected an algorithm for each histogram DIAS combines their private estimates
In the second round of the game users get to learn the basics of data dependent algorithms and under what circumstances they achieve better error rates than their data independent counterparts. The histograms to be estimated now have a different number of records (i.e., scale) and users get a first-hand knowledge on the importance of scale in the error rates of different algorithms.

In the next round, users now need to estimate the same histogram under different $\epsilon$ values where for the small value a data dependent algorithm performs best and for the high value a data independent performs the best. The main point of this round is to emphasize the importance of the privacy parameter in algorithm selection and how it is exchangeable \cite{2} with the scale parameter.

The main educational point of the last round is to introduce users to structural properties and algorithms that take advantage of these properties. More specifically, users are introduced to properties like uniformity, sparsity, and partitionality. The histograms to be answered are highly heterogeneous and users need to select algorithms that exploit different structural properties of the input. These new elements give users a valuable insight of how the structural properties of a histogram affect error rates for different algorithms.

End of the Game. Once the user has gone through all the rounds and selected an algorithm for each histogram, DIAS completes the data analysis task and assigns a score to the user which puts him on the leaderboard. The user then has the option to access Pythia’s inner workings and see exactly how Pythia made each of its choices. We achieve this goal by exposing users to both the features extracted from Pythia as well as the FAS (see Fig. 4) that Pythia used. Thus, users have a firsthand experience to learn exactly what decisions Pythia made and what features are more important in algorithm selection. Another subtle point of Pythia that users will get to see first-hand is the trade-off inherent to Pythia. The privacy budget spent for feature extraction implies that more noise will be added on the release step, but on the other hand feature extraction leads to a better algorithm choice which can decrease the error by an order of magnitude and thus have improvement on the performance. In the case that a user outperforms Pythia, we hope to have a constructive discussion on their insight for algorithm selection and what features they used in their decision making.

Our goal is that by the end of the game, both privacy experts and non-experts alike will have increased their knowledge on differential privacy and more importantly will feel even more confident in applying differentially private algorithms in their own applications.

5. REFERENCES

