Overlook: Differentially Private Exploratory Visualization for Big Data

Pratiksha Thaker
Stanford University
prthaker@stanford.edu

Mihai Budiu
VMware Research
mbudiu@vmware.com

Parikshit Gopalan
VMware Research
pgopalan@vmware.com

Udi Wieder
VMware Research
uwieder@vmware.com

Matei Zaharia
Stanford University
matei@cs.stanford.edu

ABSTRACT

Data exploration systems that provide differential privacy must manage a privacy budget that measures the amount of privacy lost across multiple queries. One effective strategy to manage the privacy budget is to compute a one-time private synopsis of the data, to which users can make an unlimited number of queries. However, existing systems using synopses are built for offline use cases, where a set of queries is known ahead of time and the system carefully optimizes a synopsis for it. The synopses that these systems build are costly to compute and may also be costly to store.

We introduce Overlook, a system that enables private data exploration at interactive latencies for both data analysts and data curators. Overlook enables fast computation of private synopses using the idea of a "virtual synopsis," which represents a potentially large private synopsis implicitly using a pseudorandom function that is evaluated on demand. Overlook uses a synopsis that is simple but fast and has accuracy comparable to other state-of-the-art synopses, requiring 0.5 seconds or less to generate a histogram over 60 GB of data – only 2.5× slower than the equivalent public histogram. Together, these allow Overlook to provide a rich, interactive, and *fast* visual query interface that allows users to explore private data with minimal storage overhead – tens of bytes – on the server.

1 Introduction

Privacy has become a key issue for all organizations that collect personal data, from companies to government entities. As a result, a number of systems [11, 5, 9] have emerged that work towards making differential privacy practical and accessible to these users.

Unfortunately, these systems are currently challenging for organizations to configure and use, in part due to the privacy budget that determines how many queries can be made to a dataset privately, and with what accuracy. At a high level, current DP systems fall into two categories. Systems such as PINQ [11] and PSI [5] ask *users* to select a privacy budget, ε, for each query they execute. This requires non-expert users to reason about tradeoffs between the value of ε and the resulting accuracy. On the other hand, systems such as PrivateSQL [9] generate a *synopsis* data structure that can answer all queries within a specific class of queries, given a total privacy budget ε. These systems are more suitable for exploratory analysis and for public access, but they are also

Figure 1: The Overlook user interface allows users to interact with visualizations in a natural, useful way, without having to reason about privacy loss budgets.

challenging to use. Constructing the synopsis requires solving a time-consuming optimization problem to minimize the error it will produce for a specific query workload, requiring several minutes for common mechanisms such as DAWA [10] and MWEM [6]. This time can become prohibitive when generating a large number of synopses, for instance when a data curator is evaluating many parameter settings to find one that will deliver good utility for a dataset. Moreover, the synopsis can consume a large amount of space, scaling with the size of the underlying domain, making it costly for large datasets.

In this paper, we present Overlook, a system that makes differential privacy practical and fast for one of the most common types of data analysis: visual data exploration. Overlook provides a natural interface for users to explore data through interactive charts, such as histograms and heat maps, at interactive latencies. In addition, Overlook's *curator interface* allows data *curators* to interactively explore different settings of privacy parameters prior to publishing the data.

The key idea in Overlook is a "virtual synopsis" data structure that compresses the representation of a synopsis into just a few bytes using a pseudo-random function (PRF). Rather than evaluating and storing an entire synopsis ahead of time, Overlook can use this virtual synopsis to compute noise corresponding to a particular query on the fly, with

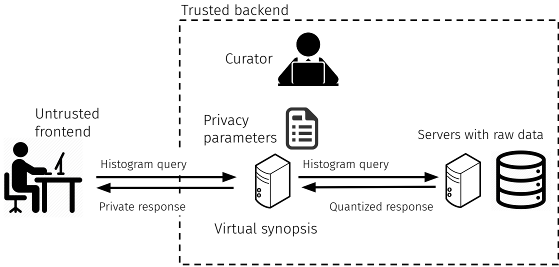


Figure 2: Overlook architecture.

minimal performance overhead. Thanks to this design, both the data curator and data analysts can run queries at a similar speed to their existing query engine without expensive storage overheads. Moreover, adding a new dataset is as simple as generating a new PRF key: no offline optimization or synopsis generation is required.

Overlook offers a rich privacy-aware *visual query interface* built on virtual synopses. Unlike existing private visualization systems, the interface is interactive and fast: it can generate a private histogram on nearly 60 GB of data in 0.5 seconds, no more than 2.5× slower than its public counterpart. Moreover, users do not need to reason about privacy budgets: they can make unlimited queries to the private synopsis, and the amount of noise added for privacy is conveyed through intuitive confidence intervals. Overlook's frontend is based on the open source Hillview system [1], but we demonstrate that the private frontend can run on top of both Hillview as well as an unmodified SQL DBMS.

Overlook is open source at `http://github.com/vmware/hillview`[1].

2 System overview

Figure 2 shows the architecture of the Overlook system. A data analyst interacts with Overlook through a browser interface that allows them to issue queries to the Overlook root node. The root dispatches the query to the backend, applies a privacy mechanism to the returned result, and returns the private result to the user.

The root also stores relevant privacy parameters used to compute the private response to a histogram query. The trusted data curator can make changes to these privacy parameters until the dataset is published, at which point the privacy parameters as well as the dataset must become immutable. The untrusted data analyst can only access published data through the private results of histogram queries. The privacy *parameters* are assumed to be public and visible to both the data curator and the data analyst. The privacy parameters are discussed further in Section 2.2.

In Section 2.1, we describe Overlook's user-facing interface and supported visualizations. In Section refsec:curator, we describe the curator interface that data curators can use to explore privacy parameters interactively.

2.1 User interface

The data curator and data analyst both access Overlook through interfaces that are extensions of the Hillview data visualization system [1], which provides a browser interface for interacting with charts and data. The result of a histogram query is displayed as an interactive plot. Additional queries can be made by zooming in using the mouse by selecting an interval, which issues a new histogram query to the backend.

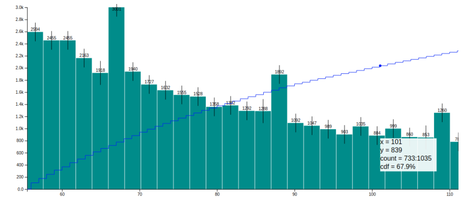


Figure 3: Histogram plot with CDF curve overlaid. Confidence intervals are plotted for each bar, and the count is displayed as a range rather than a single value.

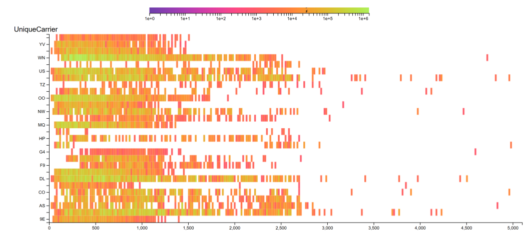


Figure 4: Heatmap on two columns. The color shows the count for each combination of values. Values with low confidence are hidden.

Note that, while the *frontend* is an extension to Hillview, Overlook can be used with any backend that supports count queries.

2.1.1 Supported visualizations

Overlook's main primitive is a histogram query. This primitive can be applied to create a variety of useful visualizations:

- Histogram queries over a column (with numeric or categorical data). The visual presentation can be a bar chart with confidence intervals, as shown in Figure 3, or, for example, a pie chart.
- Cumulative distributions functions (CDF) over a column (numeric or categorical). Figure 3 shows a histogram plot with an overlaid CDF curve.
- Histogram queries over a pair of columns, each of which can be either numeric or categorical. This can be visually presented as a heat map as in Figure 4, or for example a trellis plot of 1-dimensional histograms.

One important feature of Overlook is that it displays estimates of *uncertainty* about the data. For 1-dimensional histograms, this is in the form of 99% confidence intervals.

In addition to histograms, Overlook supports releasing certain useful counts ("degenerate histograms") privately: the number of elements and NULL values in a column, and the count of distinct values in a column.

2.2 Curator interface

The data curator's job is to decide which columns and pairs of columns will be released privately, and to then decide the privacy level for each of those data releases. Overlook's curator UI helps the data curator make these decisions.

For each set of columns that is to be released privately, the curator must specify a corresponding *privacy policy*.

[1] Overlook's UI is built on the Hillview frontend, and as such has been merged into the Hillview codebase. Documentation on the privacy-specific features can be found at `https://github.com/vmware/hillview/blob/master/privacy.md`.

This policy provides Overlook with information about public values that can be used in the histogram as well as parameters that are used to instantiate the privacy mechanism. The curator's view of the Overlook UI allows the curator to edit these policy settings and generate sample charts on a dataset before it is published.

In particular, the curator can specify:

1. The *privacy level* ε for each set of columns to be released.
2. The *data range* for each histogram, i.e. the minimum and maximum value to be displayed. These must be public and uncorrelated with the raw data. For example, for a histogram over income, the minimum may be 0 and the maximum 1 million.
3. A *quantization* for each histogram: a specification of bucket boundaries. These must be public and uncorrelated with the data. For example, for a histogram over names, the bucket boundaries may be the letters 'A' through 'Z', which leak no information about any individual's name.

While specifying a policy for *every* such histogram may be impractical, Overlook provides some useful default values for the convenience of the curator. While curators should be careful to choose parameters that are public and independent of the data, we note that the curator's decisions may nevertheless leak information because they are made based on the true underlying dataset. Devising methods to add differential privacy to this kind of human-in-the-loop parameter selection is an interesting avenue for future work.

3 Algorithmic Background

In this section, we review the mechanism Overlook uses for releasing private histograms. In Section 4, we describe how the synopsis mechanism is implemented *efficiently* in Overlook.

We use the standard definition [4] of pure ε-differential privacy in our work. Overlook operates over flat tables, but can also operate over joins that are materialized in advance.

3.1 Synopsis mechanism

The queries that Overlook targets are one- and two- dimensional histogram queries. To generate synopses for these classes of queries, Overlook uses a mechanism called the *hierarchical histogram* [8, 3]. At a high level, the hierarchical histogram builds a *tree* such that nodes higher in the tree correspond to progressively larger contiguous intervals in the domain. Each internal node of the tree corresponds to an interval of the histogram that is the union of its b children, and the mechanism adds noise with scale $\mathrm{Lap}(\log_b(m)/\varepsilon)$ to each internal node. For such a tree with branching factor b, an arbitrary interval of size t can be computed by taking the union of only $(b-1)\log_b(t)$ nodes: the number of noise variables now scales logarithmically, rather than linearly, in the interval size.

A multidimensional rectangle query can be computed by taking the Cartesian product of its decomposition in each axis.

Overlook also supports releasing certain counts, such as counts of `NULL` or missing values, privately apart from the histograms. These can be made private simply by perturbing the count with noise distributed as $\mathrm{Lap}(1/\varepsilon)$ [4]. The data curator must take into account the additional privacy cost of releasing these values.

4 Virtual Synopses

An important requirement for releasing a private synopsis is that random noise is added once, when the synopsis is constructed, and must not be resampled on future queries to the synopsis. For the hierarchical histogram mechanism, this requirement naïvely would mean that Overlook would have to store a random sample for every node in the synopsis tree, a storage overhead that grows linearly in the size of the domain.

Our solution is to use a cryptographically secure pseudorandom function (PRF). Informally, a PRF guarantees that, given a small random key, the PRF output will be indistinguishable from a truly random function to a computationally-bounded adversary [2].

The hierarchical histogram mechanism associates every interval with a unique decomposition into nodes of a tree. We use this decomposition as input to the PRF in order to deterministically sample the same Laplace noise each time a particular interval is queried, rather than explicitly instantiating the entire hierarchy of intervals.

Using the PRF, we are able to reduce the storage cost of the synopsis from linear in the domain size to a small constant – in fact, only the 32 bytes required to store the key associated with a given table.

5 Evaluation

In this section, we briefly review key highlights of Overlook's performance, and demonstrate that adding differential privacy does not substantially slow down the system or change its underlying scaling properties. In particular, the overall slowdown from privacy is no greater than 2.5×. Generating noise for any given interval query takes millseconds and requires minimal memory overhead required to store the PRF key associated with a synopsis, in comparison to existing algorithms whose computation and memory requirements scale with the data or data domain size.

We benchmark Overlook on a dataset of 20 years of U.S. flights [12]. This dataset contains 14 numeric and categorical columns over a range of data distributions and domain sizes (varying from 7 to over 4000). The total dataset size is 58.2 GB.

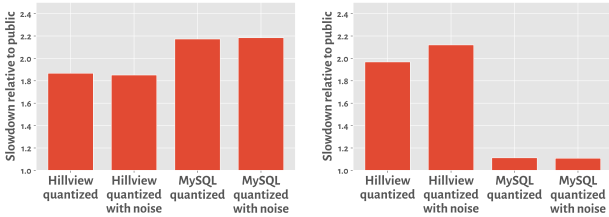
5.1 Slowdown relative to public data

We first evaluate how much differential privacy causes queries to slow down relative to queries on public data. In order to understand the slowdown, we make two measurements for each backend: first, the time required to quantize the dataset, and second, the time required to answer a quantized histogram query with noise added.

Figure 5 shows the average slowdown when plotting histograms and heat maps on the U.S. flights dataset using both the Hillview and MySQL backends. The slowdown is below 2.5× for all configurations. In all cases, the majority of the slowdown is a result of the quantization step. This is intuitive: where each data point would initially have required one operation to add it to the appropriate bucket, quantization adds an additional operation to round the point to its nearest value in the public column domain.

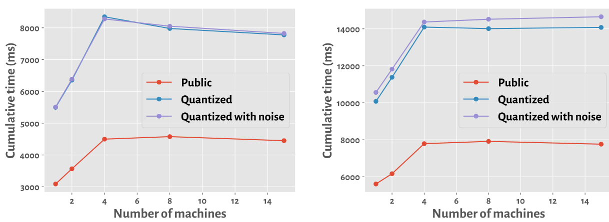
5.2 Scaling

The Overlook frontend can be used with any SQL backend. However, the Hillview distributed backend is powerful as it retains Hillview's ability to scale to large datasets.



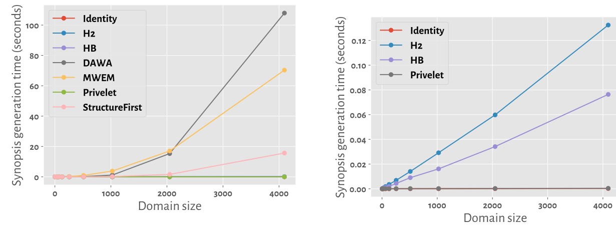
(a) Histogram slowdown. **(b)** Heatmap slowdown.

Figure 5: Slowdown relative to raw (non-private) databases for histograms and heat maps. In all cases, privacy adds at most a 2.5× performance penalty.



(a) Histogram scaling. **(b)** Heat map scaling.

Figure 6: Average time to generate histograms for columns in the flights dataset as the number of machines grows. The data size grows with the number of machines, so the runtime remains constant.



(a) Time required to generate synopses as the domain size increases. **(b)** Generation time for faster synopses.

Figure 7: Time required to generate synopses using various mechanisms, benchmarked using DPBench. MWEM and DAWA dominate in the first plot; the second plot shows that generating hierarchical histograms scales in the domain size, when not using Overlook's PRF-based construction.

We evaluate scaling using clusters of 1, 2, 4, 8,and 15 Amazon EC2 machines. The data is split equally among the machines in the cluster, so for linear scaling we expect the time required for each query to be roughly the same regardless of the number of machines. We measure time to compute charts once the data is already in memory.

In Figure 6 we show our measurements that evaluate the time breakdown for computing histograms over the U.S. flights dataset. Each point corresponds to the total time required to compute a histogram or heat map for every column or pair of columns. The overhead of privacy is the same roughly 2× overhead as in Figure 5, but privacy does not change the scaling behavior of the system at all, as expected.

5.3 Comparison to existing systems

We use DPBench [7] to evaluate the time required to generate a synopsis with the hierarchical histogram mechanism against the time required for comparable synopses. We stress that these times are not trivially comparable as DPBench is primarily an accuracy benchmark that is not optimized for performance.

We evaluate each method on a one-dimensional all-zeros dataset of increasing size, on a workload of all intervals (the workload that Overlook targets). We evaluate seven mechanisms in the literature: the baseline "identity" mechanism [4], the binary hierarchical histogram [3, 8], the hierarchical histogram with adaptive branching [8], DAWA [10], MWEM [6], Privelet [13], and StructureFirst [14].

Figure 7 shows the results of the benchmark. Figure 7a shows that MWEM and DAWA are by far the most expensive algorithms, followed by StructureFirst. The remaining algorithms run in under one second, so we plot these separately in Figure 7b. While the time required for the hierarchical mechanisms scales linearly in the data size, they are still considerably less expensive to compute than more complicated workload-aware synopses.

6 References

[1] Hillview: a big data spreadsheet. `http://github.com/vmware/hillview`. Retrieved February 2018.

[2] Dan Boneh and Victor Shoup. A graduate course in applied cryptography. *Version 0.5*, 2020.

[3] T-H Hubert Chan, Elaine Shi, and Dawn Song. Private and continual release of statistics. *ACM Transactions on Information and System Security (TISSEC)*, 14(3):26, 2011.

[4] Cynthia Dwork. Differential privacy. *Encyclopedia of Cryptography and Security*, pages 338–340, 2011.

[5] Marco Gaboardi, James Honaker, Gary King, Kobbi Nissim, Jonathan Ullman, and Salil P. Vadhan. PSI (Ψ): a private data sharing interface. *CoRR*, abs/1609.04340, 2016.

[6] Moritz Hardt and Guy N Rothblum. A multiplicative weights mechanism for privacy-preserving data analysis. In *2010 IEEE 51st Annual Symposium on Foundations of Computer Science*, pages 61–70. IEEE, 2010.

[7] Michael Hay, Ashwin Machanavajjhala, Gerome Miklau, Yan Chen, and Dan Zhang. Principled evaluation of differentially private algorithms using dpbench. In *Proceedings of the 2016 International Conference on Management of Data, SIGMOD Conference 2016, San Francisco, CA, USA, June 26 - July 01, 2016*, pages 139–154, 2016.

[8] Michael Hay, Vibhor Rastogi, Gerome Miklau, and Dan Suciu. Boosting the accuracy of differentially private histograms through consistency. *Proc. VLDB Endow.*, 3(1-2):1021–1032, September 2010.

[9] Ios Kotsogiannis, Yuchao Tao, Xi He, Maryam Fanaeepour, Ashwin Machanavajjhala, Michael Hay, and Gerome Miklau. Privatesql: A differentially private SQL query engine. *PVLDB*, 12(11):1371–1384, 2019.

[10] Chao Li, Michael Hay, Gerome Miklau, and Yue Wang. A data- and workload-aware query answering algorithm for range queries under differential privacy. *PVLDB*, 7(5):341–352, 2014.

[11] Frank D McSherry. Privacy integrated queries: an extensible platform for privacy-preserving data analysis. In *Proceedings of the 2009 ACM SIGMOD International Conference on Management of data*, pages 19–30. ACM, 2009.

[12] US Dept. of Transportation. Airline on-time performance data. `https://transtats.bts.gov/Tables.asp?DB_ID=120`. Retrieved October 2017.

[13] Xiaokui Xiao, Guozhang Wang, and Johannes Gehrke. Differential privacy via wavelet transforms. *IEEE Transactions on knowledge and data engineering*, 23(8):1200–1214, 2010.

[14] Jia Xu, Zhenjie Zhang, Xiaokui Xiao, Yin Yang, Ge Yu, and Marianne Winslett. Differentially private histogram publication. *The VLDB JournalThe International Journal on Very Large Data Bases*, 22(6):797–822, 2013.