Unleash the Power of Ellipsis: Accuracy-enhanced Sparse Vector Technique with Exponential Noise

Yuhan Liu
Renmin University of China
liuyh2019@ruc.edu.cn

Sheng Wang
Alibaba Group
sh.wang@alibaba-inc.com

Yixuan Liu
Renmin University of China
liuyixuan@ruc.edu.cn

Feifei Li
Alibaba Group
lifeifei@alibaba-inc.com

Hong Chen*
Renmin University of China
chong@ruc.edu.cn

ABSTRACT

The Sparse Vector Technique (SVT) is one of the most fundamental tools in differential privacy (DP). It works as a backbone for adaptive data analysis by answering a sequence of queries on a given dataset, and gleaning useful information in a privacy-preserving manner. Unlike the typical private query releases that directly publicize the noisy query results, SVT is less informative—it keeps the noisy query results to itself and only reveals a binary bit for each query, indicating whether the query result surpasses a predefined threshold. To provide a rigorous DP guarantee for SVT, prior works in the literature adopt a *conservative* privacy analysis by assuming the direct disclosure of noisy query results as in typical private query releases. This approach, however, hinders SVT from achieving higher query accuracy due to an overestimation of the privacy risks, which further leads to an excessive noise injection using the Laplacian or Gaussian noise for perturbation. Motivated by this, we provide a new privacy analysis for SVT by considering its less informative nature. Our analysis results not only broaden the range of applicable noise types for perturbation in SVT, but also identify the exponential noise as optimal among all evaluated noises (which, however, is usually deemed non-applicable in prior works). The main challenge in applying exponential noise to SVT is mitigating the sub-optimal performance due to the bias introduced by noise distributions. To address this, we develop a utility-oriented optimal threshold correction method and an appending strategy, which enhances the performance of SVT by increasing the precision and recall, respectively. The effectiveness of our proposed methods is substantiated both theoretically and empirically, demonstrating significant improvements up to 50% across evaluated metrics.

PVLDB Reference Format:
Yuhan Liu, Sheng Wang, Yixuan Liu, Feifei Li, and Hong Chen. Unleash the Power of Ellipsis: Accuracy-enhanced Sparse Vector Technique with Exponential Noise. PVLDB, 18(2): 187 - 199, 2024.
doi:10.14778/3705829.3705838

*Corresponding author

Proceedings of the VLDB Endowment, Vol. 18, No. 2 ISSN 2150-8097.
doi:10.14778/3705829.3705838

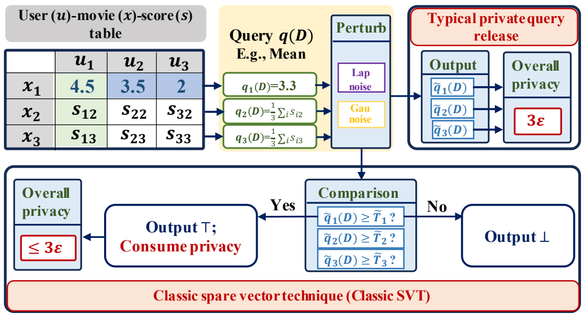

User (u)-movie (x)-score (s) table
u_1 u_2 u_3
x_1 4.5 3.5 2
x_2 s_{12} s_{22} s_{32}
x_3 s_{13} s_{23} s_{33}
Query $q(D)$ E.g., Mean
$q_1(D)$=3.3
$q_2(D)=\frac{1}{3}\sum_i s_{i2}$
$q_3(D)=\frac{1}{3}\sum_i s_{i3}$
Perturb
Lap noise
Gau noise
Typical private query release
Output
$\tilde{q}_1(D)$
$\tilde{q}_2(D)$
$\tilde{q}_3(D)$
Overall privacy
3ε
Comparison
$\tilde{q}_1(D) \geq \tilde{T}_1$?
$\tilde{q}_2(D) \geq \tilde{T}_2$?
$\tilde{q}_3(D) \geq \tilde{T}_3$?
Yes
No
Output $\top$; Consume privacy
Output $\perp$
Overall privacy
$\leq 3\varepsilon$
Classic spare vector technique (Classic SVT)

Figure 1: Classic sparse vector technique (SVT) and typical private query release on publicizing movies with the top-c highest scores. Classic SVT: after perturbing the movie score $q_i(D)$ and the predefined threshold T_i with either Laplacian or Gaussian noise, SVT compares the noisy score $\tilde{q}_i(D)$ and the noisy threshold $\tilde{T}_i$. If $\tilde{q}_i(D) \geq \tilde{T}_i$, $\top$ is output. Otherwise, $\perp$ is output. Typical private query releases: all $\tilde{q}_i(D)$ are released and sorted to obtain the noisy top-c movies.

1 INTRODUCTION

In recent days, differential privacy (DP) [13] has emerged as the predominant privacy framework adopted by numerous service providers, including Google [17, 18], Apple [8, 43], Microsoft [9, 34], *etc.*, to safeguard individuals' privacy. The sparse vector technique (SVT) [14, 15] is one of the most fundamental algorithmic tools in DP, serving as a backbone for adaptive data analysis in many applications, such as feature selection [2], stream data analysis [22], and top-c selection [31]. At a high level, SVT answers a sequence of queries on a given dataset and extracts useful information in a privacy-preserving manner. Unlike typical private query releases that directly reveal the noisy query results, *SVT discloses less information*—it outputs only a binary bit for each query, indicating whether the query result surpasses a predefined threshold (Cf. Figure 1). The major advantage of SVT over the typical private query releases is that only the positive queries (those whose results are above the predefined threshold) consume privacy, thus potentially allowing an infinite number of queries on a single dataset.

Despite the widespread usage of SVT, it still suffers from a low query accuracy [49] due to a conservative privacy analysis, resulting in a sub-optimal noise selection. Specifically, prior works in the literature naïvely approximate the privacy budget consumption of positive queries in SVT with that of differentially private query result releases (Cf. Section 2.2.2). However, since SVT discloses less information in each query (*i.e.*, only a binary bit instead of a noisy real-valued query result), this conservative privacy analysis approach overestimates its privacy risk. Consequently, SVT is constrained to use noise with large variance as in typical query releases for threshold and query perturbation, such as Laplacian [31] or Gaussian [49] noise. Such noise types then lead to an excessive noise injection and a sub-optimal query accuracy.

Motivated by this, our first contribution is providing a new privacy analysis (Cf. Theorem 2), which captures the *less informative nature of SVT* by precisely computing the privacy loss of the private comparison between thresholds and query results. Informally speaking, our analysis demonstrates that any noise whose cumulative distribution function satisfies the Lipschitz condition can be applied to SVT, allowing the use of noise types with smaller variance under the same privacy level. Based on our analysis, we identify the exponential noise, previously deemed non-applicable to SVT, as the optimal choice among all considered noise types. Exponential noise satisfies Theorem 2 tightly and benefits from a smaller variance compared to the others (Cf. Figure 4).

However, applying the exponential noise to SVT is challenging due to the bias introduced to the query results. The naïve bias correction method, which subtracts the expected value of random noise from the noisy query results, yields a sub-optimal performance. This is primarily because SVT requires correcting each noisy query result individually, whereas the naïve correction method is designed for correcting the aggregation of noisy query results [17]. When random noises are aggregated (as in the aggregation of noisy query results), their summed value approximates the expected value. In contrast, individual noisy query results are less concentrated around their expectations, resulting in less accurate corrections (Cf. Section 4.3.1).

To address this challenge, our second contribution involves developing an optimal threshold correction method[1] and an appending strategy, which effectively mitigate introduced bias and significantly enhance query accuracy. Instead of correcting the bias of each query result individually, our method focuses on optimizing the precision of the output binary bit vector. Specifically, we search for an optimal correction term that maximizes the probability of SVT distinguishing every true positive query from the true negative ones (Cf. Equation 8). Additionally, we introduce an appending strategy to complement the threshold correction method and boost the recall of SVT. Concretely, each query with a noisy negative output is appended to the end of the query queue for another round of querying. The intuition behind this is that, true positive queries are more likely to be identified as positive queries by SVT compared to the true negative ones after multiple rounds of querying, thus increasing the recall.

In conclusion, our main contributions are summarized as follows:

(1) We provide a new privacy analysis that precisely captures the privacy loss of SVT. This analysis not only broadens the noise choices for query perturbation but also identifies exponential noise as the optimal choice among all considered ones.
(2) Building on our privacy analysis, we propose a new SVT variant with exponential noise, where an optimal threshold correction method and an appending strategy are developed to mitigate the introduced bias and boost the query accuracy.
(3) Comprehensive experiments conducted on both synthetic datasets and real-world datasets demonstrate that our proposed methods significantly increase the query accuracy of SVT by 2% ~ 50% across evaluated metrics.

[1]As we further discussed in Section 4.1, correcting the threshold is equivalent to correcting the noisy query results in SVT.

2 PRELIMINARIES

In this section, we first recall the definition (Cf. Definition 1), mechanisms (Cf. Equation 1), and properties of the differential privacy (*i.e.*, post-processing (Cf. Proposition 1) and composition (Cf. Theorem 1)). Then, we proceed to describe the sparse vector technique algorithm (Cf. Algorithm 1) and some of its predominant variants.

2.1 Differential Privacy

The differential privacy (DP) was first introduced by Dwork *et al.* [13] and recently became a *de facto* standard for privacy protection. Roughly speaking, DP ensures that the likelihood of any specific output from a random algorithm varies little with sensitive neighboring inputs. The formal definition is as follows:

Definition 1 (Differential Privacy [13]). *A randomized algorithm $\mathcal{M} : \mathbb{D} \to \mathbb{R}$ satisfies (ε, δ)-differential privacy if for any two neighboring datasets D and D', and any output $o \subseteq \mathbb{O}$*

$$\Pr[\mathcal{M}(D) \in o] \leq e^{\varepsilon} \cdot \Pr[\mathcal{M}(D') \in o] + \delta.$$

When $\delta = 0$, we say that $\mathcal{M}$ satisfies pure DP (denoted by ε-DP). Otherwise, it guarantees an approximate DP.

One of the typical methods of achieving DP is through the additive-noise mechanism, which corrupts and releases query results with additive noise randomly drawn from certain distributions [20]. That is, given a dataset D, to guarantee DP, a randomized algorithm $\mathcal{M}$ releases:

$$\mathcal{M}(D) = q(D) + X, \tag{1}$$

wherein $q(D)$ is the result of a query carried on D and X is the additive noise drawn from a probability distribution $\mathcal{N}$. The noise scale is calibrated to the sensitivity of $q(D)$ defined in the following:

Definition 2 (l_p-sensitivity). *For a real-valued query $q : \mathbb{D} \to \mathbb{R}$, the l_p-sensitivity of q is defined as:*

$$\Delta_p = \max_{D, D' \in \mathbb{D}} \|q(D) - q(D')\|_p,$$

where $\|\cdot\|_p$ denotes the l_p norm and D, D' is a pair of neighboring datasets that differ by one element.

The *Laplace mechanism* is one of the most classic additive-noise mechanisms that achieves ε-DP by letting X follow a Laplace distribution centered at 0 with a noise scale $b = \frac{\Delta_1}{\varepsilon}$, denoted as $X \sim Lap\left(\frac{\Delta_1}{\varepsilon}\right)$, where Δ_1 is the sensitivity of the query. Meanwhile, the *Gaussian mechanism* mechanism typically guarantees (ε, δ)-DP by drawing the noise X from a normal distribution $N(0, \sigma^2)$, where σ is proportional to $\frac{\Delta_2}{\varepsilon}$.

DP provides a rigorous privacy guarantee mathematically and is straightforward to achieve. Additionally, there are other two properties that contribute to the wide-ranging applications of DP. Firstly, it is immune to *post-processing*, which is formally described as follows:

Proposition 1 (Post-processing[15]). *Let $\mathcal{M} : \mathbb{D} \rightarrow \mathbb{R}$ be a randomized algorithm that is (ε, δ) differentially private. Let $f : \mathbb{R} \rightarrow \mathbb{R}$ be an arbitrary randomized mapping. Then $f \circ \mathcal{M}$ is (ε, δ)-differentially private.*

Secondly, different differentially private building blocks can be elegantly combined for designing more sophisticated algorithms, as described by the following theorem:

Theorem 1 (Sequential Composition[13]). *Let $\mathcal{M}_i : \mathbb{N}^{|\mathcal{X}|} \rightarrow \mathbb{R}_i$ be an $(\varepsilon_i, \delta_i)$-differential privacy algorithm for $i \in [k]$. If $\mathcal{M}_k : \mathbb{N}^{|\mathcal{X}|} \rightarrow \prod_{i=1}^{k} \mathbb{R}_i$ is defined to be $\mathcal{M}_{[k]}(x) = (\mathcal{M}_1(x), \ldots, \mathcal{M}_k(x))$, then $\mathcal{M}_{[k]}$ is $(\sum_{i=1}^{k} \varepsilon_i, \sum_{i=1}^{k} \delta_i)$-differentially private.*

Other composition theorems. In addition to sequential composition, more advanced composition theorems with tighter composition results are also utilized in the literature [16, 21, 23, 33, 49]. Among these, composition under *Rényi differential privacy* [35], denoted as RDP, is one of the widely adopted methods in the literature, owing to its concise and tight analysis result. RDP employs Rényi divergence as a tool for privacy measurement, and combining the RDP notion with Theorem 1 yields a tighter privacy bound. Zhu *et al.* [49] later demonstrate that SVT can be combined with RDP for improved privacy guarantees. Although it is straightforward to extend the existing composition theorems applied to SVT to our proposed SVT variants [15, 49], for ease of analysis, we derive our privacy analysis in this work based on Theorem 1. While we do not dive deep into the theoretical analysis of other composition theorems in this work, an empirical study on the performance of our proposed method under RDP composition is provided in Appendix I in the full version of this work [27] to offer further insights.

2.2 Sparse Vector Technique

2.2.1 Algorithm Construction. We begin with Algorithm 1, which outlines the basic algorithm of the sparse vector technique (SVT) [14, 15, 31, 49]. In principle, SVT works as follows. Given a sequence of queries $q_i(D)$ on a dataset D and their corresponding predefined thresholds T_i, SVT first perturbs them with random noise drawn from noise distribution $\mathcal{N}_2$ and $\mathcal{N}_1$ (Line 4, Line 5, Line 1, and Line 6), respectively. Then, it compares each noisy query result $\tilde{q}_i(D)$ with the noisy threshold $\tilde{T}_i$ (Line 7). If $\tilde{q}_i(D)$ is no smaller than $\tilde{q}_i(D)$, $\top$ is output as a positive indicator (Line 9). Otherwise, $\perp$ is output as a negative indicator (Line 14). The algorithm halts either when the number of positive outcomes reaches its maximum value c, or when the total number of queries exceeds its maximum value k_{max} (Line 12).

The indicator RESAMPLE determines whether to resample the noise for threshold perturbation after a positive query is identified (Line 11). Different variants in the literature set this indicator to different values. For instance, Dwork et al. [14] set RESAMPLE to True, whereas Lyu et al. [31] argue that the overall privacy cost is

Algorithm 1: SVT. Privately indicates if query results are above thresholds.

```
Input: $Q = \{q_1, q_2, \ldots\}$,$\Delta$,$\varepsilon_1$,$\varepsilon_2$,$c$,$k_{max}$,$T = \{T_1, T_2, \ldots\}$,
       option RESAMPLE
1  $\rho \sim \mathcal{N}_1(\varepsilon_1, \Delta)$; $n_c = 0$;$n_a$=0;
2  for $i = 1, 2, 3, \ldots, k_{max}$ do
3      $n_a = n_a + 1$;
4      $v_i \sim \mathcal{N}_2(\varepsilon_2, \Delta)$;
5      $\tilde{q}_i(D) = q_i(D) + v_i$;          // Query perturbation
6      $\tilde{T}_i = T_i + \rho$;          // Threshold perturbation
7      if $\tilde{q}_i(D) \geq \tilde{T}_i$          // Private comparison
8      then
9          Output $a_i = \top$;
10         $n_c = n_c + 1$;
11         if RESAMPLE, $\rho \sim \mathcal{N}_1(\varepsilon_1, \Delta)$;
12         Abort if $n_c \geq c$ or $n_a \geq k_{max}$;
13     else
14         Output $a_i = \perp$
15     end
16 end
```

significantly reduced by setting it to False, therefore significantly boosting the performance of SVT. Furthermore, Zhu et al. [49] alternate True and False periodically for certain applications. In this work, we provide a privacy analysis of our proposed method under both True and False settings (Cf. Theorem 3). As the empirical evaluation results for both settings demonstrate similar trends and RESAMPLE=True yields better performance in general [31], we only present the results of RESAMPLE set to True in Section 5.

2.2.2 Private Comparison. As mentioned above, to provide a rigorous DP guarantee, certain types of noise are required to ensure that for any noisy threshold $\tilde{T}_i$,

$$\frac{\Pr[\tilde{q}_i(D) \geq \tilde{T}_i]}{\Pr[\tilde{q}_i(D') \geq \tilde{T}_i]} \leq e^{\varepsilon} \tag{2}$$

holds with high probability.

Previous works [14, 31, 49] bound the probability ratio on the left of Inequality 2 by naïvely assuming that Inequality 2 is equivalent to the following:

$$\frac{\Pr[\tilde{q}_i(D) \in o]}{\Pr[\tilde{q}_i(D') \in o]} \leq e^{\varepsilon}, \tag{3}$$

where o is any possible output in the output space.

To ensure Inequality 3 hold, *Laplacian* (*Lap* $\left(\frac{\Delta f}{\varepsilon_{\varepsilon_1/\varepsilon_2}}\right)$) [14, 31] and *Gaussian* noise ($N\left(0, \sigma_{1/2}\right)$, where σ_b for $b \in \{1, 2\}$ is a function of ε_b) [49] are then adopted for both the query result and predefined threshold perturbation.

However, this privacy analysis overestimates the privacy risk in SVT, leading to an overly conservative choice of noise. Note that Inequality 2 is easier to achieve compared to Inequality 3: while Inequality 3 requires a rigorous bound on the probability ratio over every possible subset (o) in the output space, Inequality 3 splits the output space into two (*i.e.*, $\geq \tilde{T}_i$ and $< \tilde{T}_i$) and only requires a bound on the *accumulative* probability over the two parts. Thus, the high probability ratios (*i.e.*, the left side term in Inequality 3) incurred by some extreme subsets o in Inequality 3 are averaged down in Inequality 2 by those with relatively low probability ratios. In other words, let O_i be the output space of $\tilde{q}_i(D)$, which is split into two parts: $\mathcal{R} := \{z \in O_i | z \geq T_i\}$ and $\mathcal{L} := O_i \backslash \mathcal{R}$. Then, by further splitting $\mathcal{R}$ into n subsets that $\mathcal{R} = o_1 \cup, \ldots, \cup o_n$, we have the following inequality holds:

$$\frac{\Pr[\tilde{q}_i(D) \geq \tilde{T}_i]}{\Pr[\tilde{q}_i(D') \geq \tilde{T}_i]} = \frac{\sum_{j=1}^{n} \Pr\left[\tilde{q}_i(D) \in o_j\right]}{\sum_{j=1}^{n} \Pr\left[\tilde{q}_i(D') \in o_j\right]} \leq \max_{j \in [n]} \frac{\Pr\left[\tilde{q}_i(D) \in o_j\right]}{\Pr\left[\tilde{q}_i(D') \in o_j\right]}$$

Intuitively, there may be some noise distributions that, although they do not satisfy Inequality 3, can still ensure that Inequality 2 holds with high probability while introducing less distortion.

2.2.3 Advantage of SVT Over Typical Private Releases. One of the most unique properties of SVT, as well as its variants, is that only the positive outcomes incur privacy costs. Specifically, the overall privacy budget ε is independent of the value of k_{max} depends instead on the noise scale (*i.e.*, the variance of the noise) of each noisy query and the number of positive outcomes n_c.

Example. Consider a scenario where a data analyst wants to identify the top-c most popular movies (Cf. Figure 1). Each movie is rated by a group of individuals who have watched it. Directly releasing the score of each movie (*i.e.*, $q_i(D)$ in Figure 1) may reveal sensitive information about whether an individual has watched a particular movie, thus necessitating the use of DP to protect individual privacy. *Typical private query releases* perturb and release the score of all candidate movies, which incurs a privacy budget proportional to the number of all candidates n. In contrast, using *SVT* to approximate the top-c movies by only outputting the indices of first c films whose scores exceed a predefined threshold T results in an overall privacy cost proportional to c rather than n. When $c \ll n$, this approach significantly reduces the privacy cost.

Note that Figure 1 uses MEAN(D) as an example of query $q_i(D)$. In practice, $q_i(D)$ can represent any queries with real-valued answers, such as SUM(D), COUNT(D), MAX(D). Furthermore, while we use top-c selection to demonstrate the effectiveness of our proposed method, SVT can be applied to many other scenarios where queries need to be evaluated against specific criteria (predefined threshold) in a privacy-preserving manner, such as feature selection [2], steaming data analysis [22].

2.2.4 Utility Metric. The utility of SVT is measured by a specialized utility metric, namely (α, β)-accuracy [15].

Definition 3 ((α, β)-accuracy). *An algorithm which outputs a stream of answers $a_1, \ldots, \in \{\top, \bot\}^*$ in response to a stream of k queries $q_1, \ldots, q_k$ is (α, β)-accurate with respect to a threshold T if except with probability at most β, the algorithm does not halt before q_k, and for all $a_i = \top$:*

$$q_i(D) \geq T - \alpha$$

and for all $a_i = \bot$:

$$q_i(D) \leq T + \alpha.$$

Definition 3 specifies that given an error tolerance parameter α, SVT achieves a success probability of at least $1 - \beta$. Particularly, queries with results falling within the interval $[T - \alpha, T + \alpha]$ are allowed to be misclassified. For instance, if a query $q(D)$ falls within $(T, T + \alpha]$, the SVT algorithm is still considered successful even if it incorrectly classifies $q(D)$ as negative by outputting $\bot$. Specifically, it is demonstrated by prior works that the SVT algorithm, when using Laplacian noise (*i.e.*, $\mathcal{N}_1$ and $\mathcal{N}_2$ are Laplace distributions), is $(\frac{8(\ln k + \ln \frac{2}{\beta})}{\varepsilon}, \beta)$-accurate [15].

3 PRIVACY ANALYSIS REVISIT

As discussed in Section 2.2.2, it has been observed that previous privacy proofs tend to overestimate the privacy risk in SVT by bounding the privacy over all possible subsets in the output space of query results. The underlying assumption is that disclosing a binary bit is as risky as directly revealing the query result itself. However, this may not hold true: intuitively, a binary bit leaks less information than the complete query result and, therefore, should pose a lower risk.

Motivated by this, we revisit the privacy analysis of SVT, taking into account its less informative nature. We provide a new analysis result in Theorem 2. Informally speaking, our results indicate that for query perturbation, SVT poses a less stringent constraint on eligible noise distributions compared to typical private query releases, thereby accommodating a broader range of noise distributions.

Theorem 2 (Privacy of SVT). *Algorithm 1 satisfies differential privacy if for any real numbers b_1, b_2, there are two positive real numbers k_1 and k_2 such that the following inequalities hold:*

$$|\ln(f_1(x)) - \ln(f_1(x + b_1))| \leq k_1 |b_1|, \tag{4}$$

$$|\ln(1 - F_2(x)) - \ln(1 - F_2(x + b_2))| \leq k_2 |b_2|, \tag{5}$$

where $f_1(\cdot)$ and $F_2(\cdot)$ are the probability density function of $\mathcal{N}_1$ and the cumulative function of the $\mathcal{N}_2$, respectively.

We defer the proof of Theorem 2 to Appendix A in our full version [27] and provide only the key takeaways here.

Takeaways from Theorem 2. First, Theorem 2 *broadens* the range of noise distributions for query perturbation. Specifically, as indicated by Equation 11, distributions such as the exponential and Gumbel distribution are now eligible for query perturbation (*i.e.*, $\mathcal{N}_2$ in Algorithm 1), whereas these types of noise were previously considered unsuitable for SVT. Second, although Theorem 2 relaxes the constrains on noise distributions for *query result perturbation*, the eligible noise distributions for *threshold perturbation* (*i.e.*, $\mathcal{N}_1$ in Algorithm 1) remain *unchanged* compared to the previous work. This is because, despite the query results not being directly released, the predefined threshold is public information. To prevent negative queries from consuming privacy, the true threshold value, which is compared with the noisy query result, must also be obscured. Hence, the same noise distributions, such as Laplacian or Gaussian, are used in traditional private query releases.

To clarify, we summarize some of the most commonly used noise distributions for both threshold perturbation and query result perturbation in Table 1.

4 CONSTRUCTION

Based on the two crucial takeaways discussed in Section 3, we propose an enhanced SVT algorithm with improved query accuracy, referred to as SVT-Exp. This algorithm is summarized in Algorithm 2, with key changes in the algorithm design highlighted by

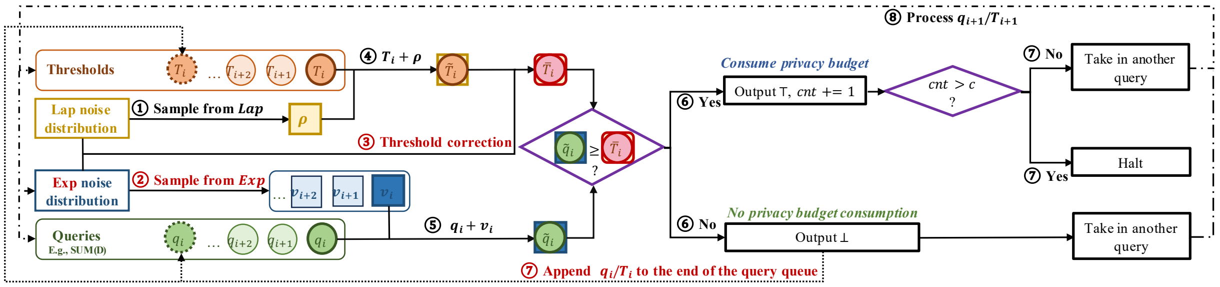


Figure 2: The overall design of Algorithm 2, with our newly proposed methods in this work emphasized with red color. The `RESAMPLE` is set to `False`. In step ③, the optimal threshold correction term r^{op} is computed based on the Laplace and exponential distribution and added to the current threshold. The threshold (step ④) and the query (step ⑤) are then perturbed with random noise sampled in step ① and ②, respectively. After comparing the noisy threshold with the noisy query in step ⑥, the algorithm outputs ⊤ and compares the number of accumulated positive outcomes (*cnt*) with a positive threshold *c* if $\tilde{q}_i \geq \bar{T}_i$. Otherwise, ⊥ is output. If `APPEND` is set to `True`, the negative query is appended to the end of the query queue for an additional round of querying. If the algorithm does not halt in step ⑦, it takes in another threshold-query pair and repeats the procedure (step ⑧).

Table 1: Feasible noise distributions for SVT

	Laplace	Gaussian	Exponential	Gumbel
Query Result	✓	✓	✓	✓
Threshold	✓	✓	×	×

underlines. Additionally, the main steps of Algorithm 2 are illustrated in Figure 2 for better understanding.

4.1 Overview

In summary, our proposed enhanced SVT algorithm introduces three key improvements:

① **The algorithm uses exponential noise for query perturbation** (Line 6). This choice is based on the fact that the cumulative probability function of exponential noise tightly satisfies Equation 11 in Theorem 2. Meanwhile, it exhibits a much smaller variance compared to other noise distributions considered, resulting in less data distortion and enhanced query accuracy.

②: **An optimal threshold correction term is computed by maximizing the (α,β)-accuracy of the SVT with exponential noise** (Line 3), and added to the noisy thresholds $\tilde{T}_i$ (Line 9). Correcting the threshold is crucial when using exponential noise (or other noise distributions centered at non-zero values) as it introduces bias into the query results[2]. Intuitively, the correction term that maximizes the (α,β)-accuracy ensures the highest success probability (*i.e.*, $1-\beta$) for SVT, thereby improving query accuracy.

③: **An appending strategy has been developed**, where noisy negative queries (*i.e.*, $q_i(D)$ for $\tilde{q}_i(D) < \tilde{T}_i(D)$) are appended to the query queue for an additional round of querying (Line 18 to Line 19). Generally speaking, while our optimal threshold correction method effectively increases the precision of SVT, the appending strategy further increases its recall. The rationale behind this strategy is that for a positive query where $q_i(D) \geq T_i$, comparing the noisy threshold $\tilde{T}_i$ with $\tilde{q}_i(D)$ multiple times increases the likelihood of $q_i(D)$ identified as a positive query.

The concrete details of our developed methods and strategy are presented in Section 4.2, Section 4.3, and Section 4.4. Additionally, the privacy guarantee and the utility guarantee of Algorithm 2 are provided in Theorem 3 and Theorem 4, respectively. The proof of

Algorithm 2: SVT with exponential noise, optimal threshold correction, and appending strategy (SVT-Exp).

```
Input: $Q = \{q_1, q_2, \ldots\}$,$\Delta$,$\varepsilon_1$,$\varepsilon_2$,$\lambda$,$c$,$k_{max}$,$T = \{T_1, T_2, \ldots\}$,$\alpha$,
       $\varepsilon$, $m$,$e$,$k$ option RESAMPLE, option APPEND.
1  $\varepsilon = \varepsilon_1 + \varepsilon_2$; $n_c = 0$; $n_a = 0$;
2  $b = \frac{\Delta}{\varepsilon_1}$;$\rho \sim Lap(b)$;
3  $\underline{r^{op} = \text{CorrectionTerm}(b, \lambda, \alpha, m, e, k)}$;      // Correction
4  for $i = 1, 2, 3, \ldots$ do
5      $n_a = n_a + 1$;
6      $\underline{\lambda = \frac{\varepsilon_2}{2c\Delta}; v_i \sim Exp(\frac{1}{\lambda})}$;      // Query perturbation
7      $\underline{\tilde{q}_i(D) = q_i(D) + v_i}$;
8      $\tilde{T}_i = T_i + \rho$;      // Threshold perturbation
9      if $\underline{\tilde{q}_i(D) \geq \tilde{T}_i + r^{op}}$      // Private comparison
10     then
11         Output $a_i = \top$;
12         $n_c = n_c + 1$;
13         $S_\top = S_\top \cup i$;
14         if RESAMPLE, $\rho \sim Lap(b)$;
15         Abort if $n_c \geq c$ or $n_a \geq k_{max}$;
16     else
17         Output $a_i = \perp$;
18         if $\underline{\text{APPEND}}$ then
19             $\underline{Q = Q \cup \{q_i\};T = T \cup \{T_i\}}$;      // Appending
20         end
21     end
22 end
```

[2]Note that correcting the threshold is equivalent to correcting the query results. In essence, as SVT compares $\tilde{q}_i - \tilde{T}_i$ with 0. Therefore, adding the correction term to the predefined threshold $\tilde{T}_i$ is the same as subtracting it from the query result $\tilde{q}_i(D)$.

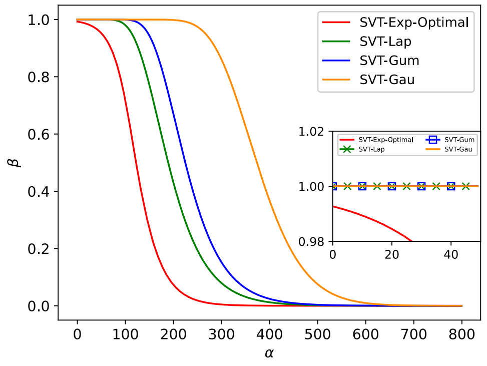


Figure 3: The (α, β)-accuracy of SVT algorithms with four different types of noise: the SVT with the exponential noise and the optimal threshold correction; SVT with the Laplacian noise; SVT with the Gumbel noise and the mean threshold correction; and SVT with Gaussian noise. Parameter k in Theorem 4 is set to 50 and the overall privacy budget $\varepsilon = 1$.

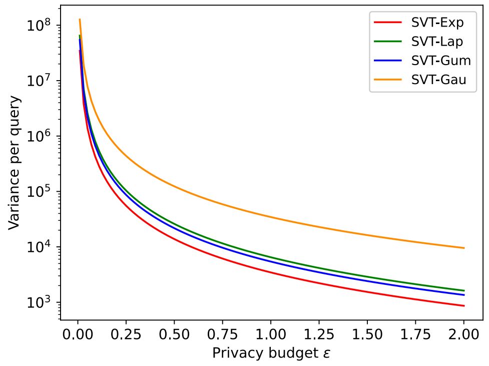


Figure 4: The variance of SVT with four different types of noise (*i.e.*, exponential noise, Laplacian noise, Gumbel noise, and Gaussian noise). The privacy budget ε varies from 0.01 to 2. c is set to 50. The y-axis is log-based. Note that the threshold correction does not affect the query variance.

these theorems is deferred to Appendix B and Appendix C in our full version [27].

Theorem 3 (Privacy guarantee). *Let c denote the number of positive outcomes output by Algorithm 2. Algorithm 2 satisfies $(\varepsilon_1 + \varepsilon_2)$-differential privacy when* `RESAMPLE` *is set to* `False`*, and $(c\varepsilon_1, \varepsilon_2)$-differential privacy when* `RESAMPLE` *is set to* `True`*, where ε_1 and ε_2 are the privacy budgets for threshold and query perturbation, respectively.*

For the utility guarantee, following the convention [15, 31], (α, β)-accuracy is adopted as the utility metric. The parameters in Algorithm 2 are set as $\Delta = 1$, $\varepsilon_1 = \varepsilon_2 = \frac{\varepsilon}{2}$ and $c = 1$ for ease of computation and comparison. As demonstrated in Theorem 4, compared to SVT algorithms where the Laplacian noise is adopted for query perturbation, Algorithm 2 demonstrates a clear advantage.

Theorem 4 (Utility guarantee). *Given any k records such that $|\{i < k : d_i \geq t - \alpha\}| = 0$ (i.e., the record above and closest to the threshold is the last one), Algorithm 2 is at least $(\frac{4(\ln k + \ln \frac{2}{\beta})}{\varepsilon}, \beta)$-accurate.*

Additionally, note that the assumptions made on parameter settings in Theorem 4 may not fully align with real-world scenarios. Therefore, we further provide a numerical utility analysis under a more practical setting in Figure 3, which shows that the value of β of Algorithm 2 is consistently below that of other baselines for a fixed α, further demonstrating the effectiveness of our method.

4.2 Query Perturbation with Exponential Noise

In this section, we justify the use of exponential noise in Algorithm 2 for the following two reasons:

First, exponential noise guarantees DP for SVT, as its cumulative function,

$$F(x; \lambda) = \begin{cases} 1 - \exp(-\lambda x) & x \geq 0 \\ 0 & x < 0 \end{cases},$$

tightly satisfies Equation 11 with $k_2 = \lambda$ for any $\lambda > 0$. However, note that the exponential distribution does not satisfy Equation 10, and therefore cannot be used for threshold perturbation. Since the Laplace distribution is one of the most commonly and frequently adopted noise distributions in SVT, we use it to perturb the predefined threshold, following convention.

Second, exponential noise yields the smallest variance on the private comparison result (*i.e.*, $\tilde{q}_i(D) \geq \tilde{T}_i + r^{op}$) in Algorithm 2 among all evaluated noises, yielding the least data distortion among all compared methods. Specifically, as also stated in [31], the variance of the private comparison result V is written as follows:

$$V = \mathsf{Var}\left(\mathsf{Lap}\left(\frac{\Delta}{\varepsilon_1}\right)\right) + \mathsf{Var}\left(\mathsf{Exp}\left(\frac{2c\Delta}{\varepsilon_2}\right)\right). \tag{6}$$

A smaller V indicates a smaller data distortion, thus a potentially higher query accuracy. Given an overall privacy budget ε, by varying w such that $\varepsilon_2 = w\varepsilon_1$ and $\varepsilon = \varepsilon_1 + \varepsilon_2$, V can be minimized for any fixed pair of noise distributions [31] (Cf. Appendix G in our full version [27]). As shown in Figure 4 where the minimal V for different $\mathcal{N}_2$ in Algorithm 2 are depicted, exponential distribution yields the smallest private comparison variance.

4.3 Optimal Threshold Correction

In this section, we elaborate on our optimal threshold correction by: (1) providing the motivations necessitating our design (Cf. Section 4.3.1); (2) detailing our correction methodology, which incorporates derivation of the success probability of SVT and computation of the optimal correction term (Cf. Section 4.3.2); and (3) presenting a numerical computation framework that generalizes our correction method to broader applications (Cf. Section 4.3.3).

4.3.1 Motivations. Though the exponential noise demonstrates properties that can theoretically boost query accuracy, applying it to SVT is challenging due to the introduced bias. Specifically, the expectation of the difference between a noisy query and its corresponding noisy threshold is written as follows:

$$E\left(\tilde{q}_i(D) - \tilde{T}_i\right) = q_i(D) - T_i + \frac{2c\Delta}{\varepsilon_2}.$$

Due to the introduction of a positive bias $\frac{2c\Delta}{\varepsilon_2}$, the algorithm is likely to mistakenly classify queries $q_i(D)$ as positives if they fall within the range $T_i \geq q_i(D) \geq T_i - \frac{2c\Delta}{\varepsilon_2}$. As a result, the query accuracy is compromised for two reasons. First, as mentioned above, the false positive rate increases. Second, as described in Algorithm 2, to limit the overall privacy budget consumption, no more than c records are allowed to be identified as positive queries. Since the false positive queries occupy these c spots, the subsequent true positive queries are unable to be output, thereby decreasing the precision.

A naïve solution is to directly subtract the bias from each $q_i(D)$ or add the bias to each T_i [29, 30, 44, 46]. However, this method yields a sub-optimal performance when applied to SVT. The primary reason is that this approach is designed for correcting the aggregation of n noisy query results, whereas our focus is on correcting each noisy query result individually. According to the law of large numbers, the mean of the aggregation with larger n tends to converge to its expectation. That is to say, as n increases, the mean of the aggregated results approaches the noise expectation more closely. Consequently, correcting the aggregate of noisy query results by subtracting the expectation often achieves satisfactory accuracy. However, since our approach targets individual noisy results (*i.e.*, $n = 1$), this method does not provide the same level of performance in SVT as it does with aggregated noisy queries.

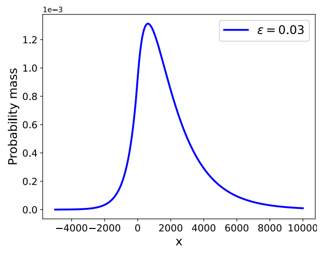
4.3.2 Methodology. Motivated by this, we have developed a new correction method that focuses on improving the precision of the output binary vector rather than correcting each individual query result. At a high level, inspired by the definition of (α, β)-accuracy (Cf. Definition 3), we compute the success probability of the SVT algorithm by multiplying the probability of each query being corrected classified. We then derive an optimal threshold correction term by maximizing this success probability. In essence, our optimal correction method increases the threshold value to enhance the precision of SVT, which is further elucidated in the analysis presented in Figure 5.

Success probability of SVT. We begin by considering a simple case where $c = 1$. Assume there are k true negative queries before the true positive query. Given a fixed error tolerance parameter α, the success probability of Algorithm 2 with a threshold correction term r is defined as follows:

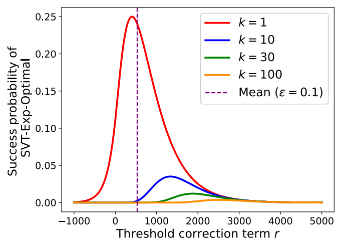
$$p(r) = \prod_{i=1}^{k} \Pr\left[\tilde{q}_i(D) - \alpha < \tilde{T}_i + r\right] \cdot \Pr\left[\tilde{q}_j(D) + \alpha \geq \tilde{T}_j + r\right], \tag{7}$$

where $\tilde{q}_{i/j}(D) = q_{i/j}(D) + \mathsf{Exp}(\frac{1}{\lambda})$, and $\tilde{T}_{i/j} = T_{i/j} + \mathsf{Lap}(b)$. Note that here we abuse the notion $\mathsf{Exp}(\cdot)$ and $\mathsf{Lap}(\cdot)$ to denote the random variables drawn from exponential distribution and Laplace distribution, respectively.

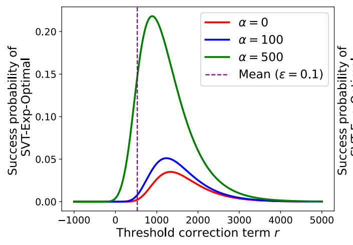
Takeaways from Equation 7. First, Equation 7 is directly related to the (α, β)-accuracy. Specifically, for a fixed α, the value of β with a threshold correction term r is given by $1 - p(r)$. Second, by maximizing $p(r)$, we maximize the query accuracy of SVT, which in turn improves empirical performance. Third, Equation 7 can be easily extended to scenarios where $c > 1$ by: ① dividing all queries into subroutines, each containing exactly one positive query; and ② multiplying the success probability of each subroutine.



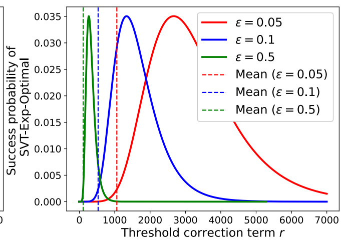
(a) The probability distribution of the random variable $Z = X - Y$ with $X \sim \mathrm{Lap}(\frac{1}{\varepsilon_1})$ and $Y \sim \mathrm{Exp}(\frac{1}{\varepsilon_2})$. The parameters α and k are set to 0 and 10, respectively.



(b) $p(r)$ with varying k. The parameter α is set to 0 and the overall privacy budget ε is set to 0.1.



(c) $p(r)$ with varying α. The overall privacy budget ε is set to 0.1 and the parameter k is set to 10.



(d) $p(r)$ with varying overall privacy budget ε. The parameters α and k are set to 0 and 10, respectively.

Figure 5: The numerical analysis of Equation 8. Dashed lines are the mean of the exponential noise distribution. Each privacy ε has a corresponding mean.

However, Equation 7 cannot be directly adopted as it is data-dependent, which could lead to privacy leakage. To address this issue, we adopt a worst-case assumption by setting $q_{i/j}(D) = T_{i/j}$. Thus, the success probability $p(r)$ can be expressed as:

$$p(r) = \left(\Gamma\left(r + \alpha\right)\right)^k \cdot \left(1 - \Gamma\left(r - \alpha\right)\right), \tag{8}$$

where $\Gamma(\cdot)$ is the cumulative distribution function (CDF) of the random variable $Z = X - Y$. Here X and Y are random variables drawn from distribution $\mathsf{Exp}\left(\frac{1}{\lambda}\right)$ and $\mathsf{Lap}\left(b\right)$, respectively. The shape of the probability density function (PDF) of Z is illustrated in Figure 5(a), and the explicit expression of $p\left(r\right)$ is detailed in Appendix D in our full version [27].

Optimal correction term. After demonstrating that the maximum value of $p(r)$ exists (Cf. Appendix E in our full version [27]), the optimal threshold correction is obtained by:

$$r^{op} = \arg\max p(r). \tag{9}$$

To offer a better understanding, we perform a detailed analysis of Equation 8 and Equation 9 (Cf. Figure 5), and the key findings are summarized as follows:

First, as shown in Figure 5(b), an increasing number of negative queries (k) leads to a larger threshold correction term (r^{op}). As k grows, SVT is more likely to halt before the last true positive query, which is referred to as 'early halting'. Consequently, a larger correction term is necessary to mitigate early halting, thereby enhancing the precision of the SVT algorithm.

Second, as shown in Figure 5(c), an increase in the error tolerance parameter α results in a higher success probability and a smaller threshold correction term r^{op}. The rationale is that a larger α allows more false negative or false positive queries during the query process. Consequently, a smaller threshold correction term is sufficient to achieve a high success probability. Note that α is a hyperparameter chosen by the data analysts. For generality, this work defaults to $\alpha = 0$, irrespective of specific data distributions.

Third, as shown in Figure 5(d), an increase in the privacy budget results in a smaller threshold correction term. This is because a larger privacy budget leads to lower noise variances, which reduces data distortion and results in noisy query results that are more tightly concentrated around their expectations. Consequently, a smaller threshold correction term is adequate to prevent the early-halting issue and achieve an optimal success probability.

4.3.3 Numerical Computation Framework. To improve the scalability of the optimal threshold correction method, we further propose a numerical computation framework, which is summarized in Algorithm 3 and Algorithm 4. Instead of deriving the analytical formula for $\Gamma(\cdot)$ in Equation 8, we first discretize the noise distributions $\mathcal{N}_1$ and $\mathcal{N}_2$ used in Algorithm 2. We then convolve these discretized distributions using the Fast Fourier Transform (FFT) to obtain the numerical representation of Γ. The optimal threshold correction term, $\bar{r}^{op}$, is determined by maximizing $\bar{p}(r)$, which is calculated based on this numerical $\Gamma(\cdot)$. Hereinafter, we use the combination of Laplace and exponential distributions as an example to illustrate the concrete steps of our framework. First, to convert the continuous noise distributions into discretized sequences, we define a boundary B within which most of the probability mass (*e.g.*, $1 - e$ as indicated in Line 1 of Algorithm 3) is concentrated. Next, in Algorithm 4, we discretize the event space within B into u chunks, with each chunk having a mesh size m (Line 1). The events in each chunk are aggregated into a new event with a probability mass h_i (Line 3). Additionally, the probability mass of events exceeding B is assigned to the positive infinity bracket, while events below $-B$ are placed in the negative infinity bracket (Lines 5 and Line 6 in Algorithm 4). Subsequently, we use Fast Fourier Transform (FFT) [36] to compute the discretized distribution $\bar{\Gamma}$ by convolving the discretized Laplace distribution $\bar{L}$ and the discretized exponential distribution $\bar{E}$, where we define that $\infty + x = \infty$ for any $x \in \mathbb{R}$. Finally, the correction term $\bar{r}$ is determined using $\bar{\Gamma}$ based on Equation 9.

Algorithm 3: CorrectionTerm: Numerical threshold correction.

```
  Input: $b, \lambda, \alpha, m, e, k$
1 $B = F^{-1}_{Exp}(1 - e)$;// Boundary
2 $\bar{L}$ = Discretizer($Lap(b), m, B$);; // Discretize Laplace distribution
3 $\bar{E}$ = Discretizer($Exp(\frac{1}{\lambda}), m, B$);; // Discretize exponential distribution
4 $\bar{\Gamma}$ = CONVOLVE($\bar{L}, \bar{E}$);; // Convolution with FFT
5 Compute $\bar{r}^{op}$ that maximize $\left(\bar{\Gamma}(\bar{r} + \alpha)\right)^k \cdot (1 - \bar{\Gamma}(\bar{r} - \alpha))$;
6 return $\bar{r}^{op}$;
```

Algorithm 4: Discretizer

```
  Input: CDF, $m, B$
1 $u = \frac{B}{m-1}$;
2 for $i$=-m+1 to m-2 do
3     $h_i$ = CDF($(i + 1) \cdot u$) − CDF($i \cdot u$);
4 end
5 $h_{-\infty}$ = CDF($(1 - m) \cdot u$);
6 $h_{+\infty}$ = 1 − CDF($m - 1 \cdot u$);
7 $\bar{D} = \{h_{-\infty}, h_i$ for $i$ in $(-m, m) \cap \mathbb{Z}, h_{+\infty}\}$;
8 return $\bar{D}$;
```

The primary advantage of this numerical computation framework is its strong scalability to various types of noise distributions. While we can derive the explicit formula for $\Gamma(\cdot)$ when using Laplace and exponential distributions for query and threshold perturbation, respectively (Cf. Appendix D in the our version [27]), deriving such formulas becomes complex when combining other distributions, such as Gaussian and exponential. This numerical framework allows privacy practitioners to apply the optimal threshold correction method to any noise distributions. It is important to note that our numerical framework may introduce additional computation costs due to the discretization and convolution steps. To provide further insights, we analyze the trade-off between the running time of our algorithm and the estimation accuracy of $\bar{p}(r)$ and $\bar{r}^{op}$ in Appendix K in the full version [27]. In summary, our analysis demonstrates that the proposed method can achieve highly accurate estimations with minimal additional computation cost.

4.4 Appending Strategy

As discussed earlier, our threshold correction method enhances the performance of SVT by increasing the value of the threshold, which in turn improves the precision of the algorithm. However, a higher threshold can also result in a reduced recall. This occurs because true positive queries with relatively small results are less likely to exceed the adjusted threshold.

To further enhance recall, we propose an appending strategy. As shown in Figure 2 Step ⑦ and Algorithm 2 from Line 18 to Line 19, each query identified as negative by the algorithm is appended to the end of the query queue for another round of querying. In settings where there is an infinite number of queries (*e.g.*, streaming data analysis, where new data and queries continuously arrive), these queries may be randomly inserted back into the queue.

The rationale behind this strategy is twofold. First, as stated in Theorem 3, re-querying the outputs identified as negative incurs no additional privacy cost. Second, conducting multiple rounds of querying increases the likelihood of correctly identifying true positive queries that were misclassified initially. Simultaneously, it also enlarges the probability gap between positive queries correctly identified as positive and negative queries mistakenly classified as positive. Concretely, for each query $q_i(D)$, the probability of it identified as positive in each round is

$$p_i = \Pr[q_i(D) + \mathsf{Exp}(\frac{1}{\lambda}) \geq T_i + r + \mathsf{Lap}(\mathsf{b})].$$

After t rounds of querying, this probability increases to tp_i. Also note that the larger the difference $q_i(D) - T_i$, the higher the p_i. Therefore, after multiple rounds, the probability of $q_i(D)$ being identified as positive is $t\left(p_i - p_j\right)$ higher than for $q_j(D)$, where $q_i(D) - T_i \geq q_j(D) - T_j$.

5 EVALUATION

Before presenting our main results in Section 5.5, we detail the used datasets in Section 5.1, the adopted utility metrics in Section 5.2, the compared baselines in Section 5.3, and the selected parameters for our experiments in Section 5.4.

5.1 Datasets and Implementations

This work uses six different datasets: three real-world and three synthetic. Detailed information is provided in Table 2.

Table 2: Dataset details. # · denotes the number of ·, and the 'Threshold' is the predefined threshold we use in SVT for each dataset.

	# of records	# of items	Threshold	Type
Binary	-	10,000	500	synthetic
Zipf	-	10,000	200	
T40I10D100K [39]	100,000	942	11,850	
BMS-POS [48]	515,597	1,657	13,600	real-world
Kosarak [1]	990,002	41,270	10,500	
Adult [3]	48,843	123	200	

For the real-world datasets and the T40I10D100K dataset, each item's score is based on its frequency across records. Specifically, for each item I_i in the dataset, the score $s_i = \sum_{j=1}^{n} \mathbb{1}(I_j^i = 1)$, where I_j^i is an indicator of whether the record R_j contains item I_i, and n is the total number of records. For the remaining synthetic datasets, scores are assigned directly. In the Binary dataset, each positive query is assigned a score of 1,000, while negative queries receive a score of 0. In the Zipf dataset, each item's score is proportional to $\frac{1}{i}$, with the score calculated as $s_i = \frac{1}{i} \times 10,000$. We use m to denote the total number of items. Additionally, Figure 6 plots the item scores for each dataset for further insights.

All experiments are conducted on a laptop (6-core Intel Core i7 CPU at 2.2 GHz with 16-GB RAM).

5.2 Queries and Utility Metric

5.2.1 Queries. The effectiveness of our proposed method is general but is validated here within the top-c selection problem, a common application for SVT [31, 49]. In this scenario, we use SVT to approximately query items with the top-c highest scores in each dataset.

Specifically, we first shuffle the items randomly, then query each item's score (*i.e.*, $q_i(D) = s_i$) and compare it to the threshold T. If $q_i(D) \geq T$, the corresponding query index i is output. The algorithm halts either when c indices is output or the number of queries

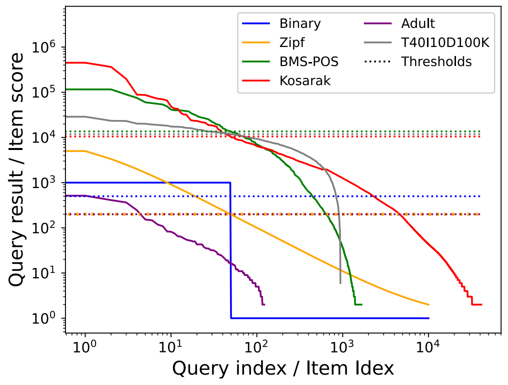


Figure 6: The scores of items of 6 datasets. All the items are in descending order based on their scores, which are plotted in a log-based manner for clarity. Dashed lines in Corresponding colors are the predefined thresholds we adopt in this work.

reaches a maximum limit k_max. The choice of T is crucial for query accuracy, but determining an appropriate threshold is beyond this work's scope. Various methods for threshold determination are discussed in the literature [6, 26].

5.2.2 Utility Metrics. We evaluate SVT's performance on the top-c selection problem using two primary metrics: **F1-score** [19] and normalized cumulative rank (NCR). The F1-score is computed as: $F1 = \frac{2\cdot\text{precision}\cdot\text{recall}}{\text{precision}+\text{recall}} = \frac{2TP}{2TP+FP+FN}$, where TP is the number of true positive queries, FP is the number of false positive queries,and FN is the number of false negative queries. However, the F1-score is more suited for unordered settings, where missing a top result incurs the same penalty as missing a lower-ranked result [31]. To address this, we also use the Normalized Cumulative Rank (NCR) [31]. In NCR, each query q_i is assigned a rank score defined as follows: the top query (*i.e.*, top-1) receives a score of c, the next receives $c - 1$, and so on [31]. Queries below the threshold receive a score of 0. The total score for positive outcomes is then normalized to the range $[0, 1]$ by dividing by $\frac{c(c+1)}{2}$, the maximum possible score. Since the results for the F1-score demonstrate similar trends as NCR, detailed F1-score results are provided in Appendix H in the full version [27]

5.3 Baselines

Table 3 lists all the methods compared in this work. We evaluate our method, which uses exponential noise for query perturbation, against **SVT-Lap** [31], **SVT-Gau** [49], and **SVT-Gum**. While SVT-Lap and SVT-Gau are two of the most frequently used SVT variants in the literature, SVT-Gum is a new variant proposed in this work. SVT-Gum meets the constraints in Theorem 2 but has a slightly larger variance compared to SVT-Exp. More details about SVT-Gum are provided in Appendix F in the full version [27]. Additionally, to provide more insights, we also compare our method with the 'upper bound' of query accuracy, which is obtained by directly ranking the noisy query results perturbed with random noise drawn from $\mathsf{Exp}\left(\frac{\Delta}{\varepsilon_2}\right)$. Regarding threshold correction, we compare SVT-Exp with optimal threshold correction to SVT-Exp with no threshold correction and to SVT-Exp with mean correction (*i.e.*, the naïve solution in Section 4.3.1).

Table 3: Baseline methods. Lap, Exp, Gau, and Gum refer to Laplace, exponential, Gaussian, and Gumbel distribution, respectively. 'Optimal' represents our optimal threshold correction method, while 'Mean' represents the naïve correction method by subtracting the mean of the noise.

	Noise for query	Noise for threshold	Threshold correction
SVT-Exp upper bound	-	Exp	-
SVT-Exp (optimal) (Alg. 2)	Lap	Exp	Optimal
SVT-Exp (mean)	Lap	Exp	Mean
SVT-Exp (no)	Lap	Exp	No
SVT-Gumbel	Lap	Gum	Mean
SVT-Lap	Lap	Lap	No
SVT-Gau	Gau	Gau	No

5.4 Parameter Selection

This work involves several privacy parameters: the failure rate δ, the overall privacy budget ε, the privacy budget for threshold perturbation (*i.e.*, ε_1), and the privacy budget for query perturbation (*i.e.*, ε_2). For SVT-Gau, δ is set to $\frac{1}{n}$, following convention [15, 28, 29], while in other settings, δ is set to 0. As also mentioned in Section 4.2, ε is divided into ε_1 and ε_2 such that $\varepsilon_2 = w\varepsilon_1$ and $\varepsilon = \varepsilon_1 + \varepsilon_2$. The parameter w is computed by minimizing Equation 6 [31]. Table 4 lists w for different baselines, with its proof provided in Appendix G in the full version [27].

Table 4: Optimal Privacy Allocation. While ε is the overall privacy budget consumption, ε_1 and ε_2 are the privacy budget for threshold perturbation and query perturbation, respectively. w is an indicator of privacy budget allocation.

	SVT-Exp	SVT-Gum	SVT-Lap	SVT-Gau
ε		$\varepsilon = \varepsilon_1 + \varepsilon_2,\ \varepsilon_2 = w\varepsilon_1$		
w	$\left(\sqrt{2}c\right)^{2/3}$	$(\frac{\pi c}{\sqrt{3}})^{2/3}$	$(2c)^{2/3}$	$(2c)^{2/3}$

5.5 Evaluation Results

Hereinafter, we demonstrate the effectiveness of our proposed methods through experiments on six datasets, with an in-depth analysis of the evaluation results. Our main results, presented in Figure 7, show a significant advantage of Algorithm 2 over other baselines. The correctness and effectiveness of our optimal threshold correction method are shown in Table 5 and Figure 7. Finally, the effectiveness of our appending strategy and the trade-off it yields between efficiency and query accuracy is illustrated in Figure 8.

5.5.1 Effectiveness of Algorithm 2. First, as illustrated in Figure 7, our proposed method (*i.e.*, SVT-Exp (optimal correction), shown with red-circle line) significantly outperforms others baselines by up to 50% on NCR across all tested privacy regions and datasets. Moreover, compared to other baselines, the performance of our proposed method closely matches the empirical SVT-Exp upper bound, further demonstrating its effectiveness. Second, our method shows a particular larger advantage when the gaps between predefined threshold and query results are relatively large (Cf. Figure 7(b) and Figure 6). This is because our threshold correction better filters out true positive queries far above the threshold, while our appending strategy effectively distinguishes smaller true positives from true negatives far below the threshold. Third, all compared variants, including ours, demonstrate higher NCR on datasets where the gaps between the threshold and query results are large even under smaller privacy budget (*e.g.*, $\varepsilon = 0.05$ on Kosarak), as these datasets are generally more robust to noise. Fourth, the NCR of other baselines decreases from SVT-Lap, SVT-Gumbel, to SVT-Gau, which aligns with our theoretical analysis in Figure 3 and Figure 4.

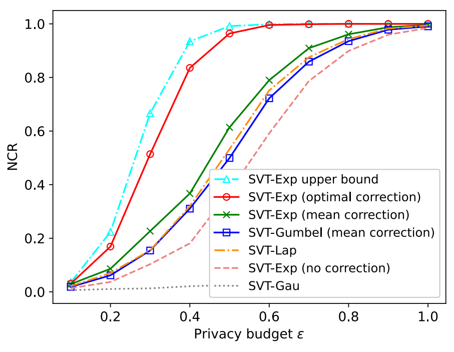
Table 5: Comparison of the threshold correction term between the optimal threshold correction method and the mean correction method, where $c = 50$ and $\alpha = 0$.

ε	0.01	0.05	0.1	1	2
Optimal	35450.45	7147.15	2803.30	280.78	138.89
Mean	5332.08	1046.42	523.21	52.32	26.16

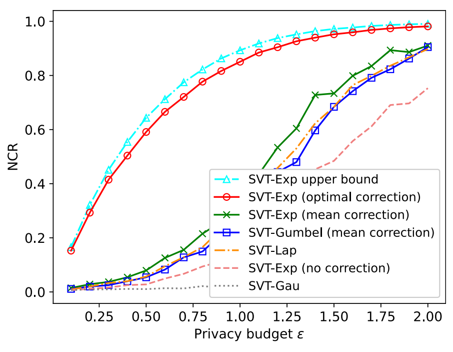
5.5.2 Effectiveness of the Optimal Threshold Correction. We compare our SVT-Exp (optimal correction) with SVT-Exp (mean correction) and SVT-Exp (no correction) in Figure 7, while Table 5 presents the values of optimal correction term r^{op} with different values of ε. First, as shown in Table 5, the optimal term is usually larger than the mean of the injected noise, aligning with our analysis in Section 4.3.2. Second, SVT-Exp without correction demonstrates relatively low NCR even compared to, *e.g.*, SVT-Lap, highlighting the need for a threshold correction method when using exponential noise in SVT. Third, SVT-Exp with our optimal correction terms drastically outperforms SVT-Exp with mean correction, which slightly outperforms the other baselines. This demonstrates the effectiveness of both exponential noise and optimal threshold correction methods.

5.5.3 Effectiveness of the Appending Strategy. To demonstrate the effectiveness of our appending strategy, we compare the performance of all our baselines at the same overall privacy budget consumption with varying numbers of traverse (*i.e.*, the number of times a query with the noisy negative outcome compared with the threshold)[3]. As shown in Figure 8, our proposed method performs better across most datasets under all tested traverse numbers. Notably, our method shows slight inferiority on the T40I10D100K dataset under a very small number of traverses. One possible reason is the relatively small gap between positive query results and the predefined threshold on T40I10D100K (Cf. Figure 6): due to our relatively high optimal threshold correction term, more queries are needed to filter out the true positive queries, as explained in

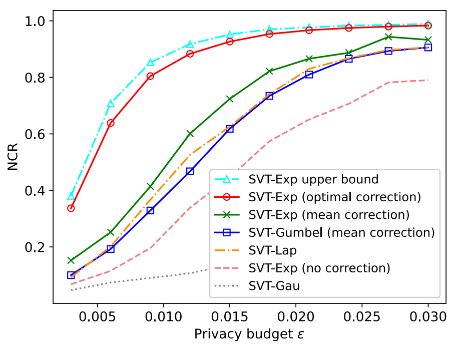
[3]For fairness, under each number of traverses, we ensure that each baseline consumes the same amount of the privacy budget before comparing their performance. Hence, the overall privacy budget consumption varies with a different number of traverses.



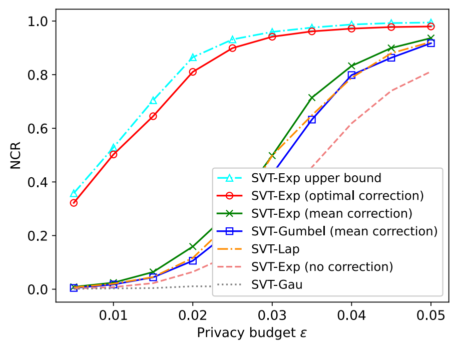
(a) Binary dataset



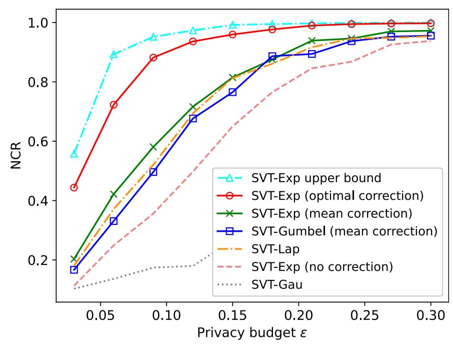
(b) Zipf dataset



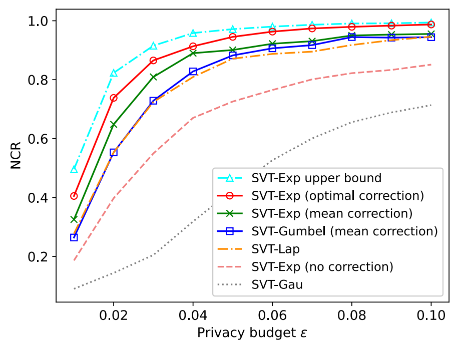
(c) BMS-POS dataset



(d) Kosarak dataset



(e) Adult dataset



(f) T40I10D100K dataset

Figure 7: NCR on six datasets with $c = 5$ for Adult dataset and $c = 50$ for the remaining datasets, $\alpha = 0$, $k = \lfloor \frac{m}{c} \rfloor$, `RESAMPLE=False`, and `APPEND=True`. Sequential composition theorem is adopted for computing ε.

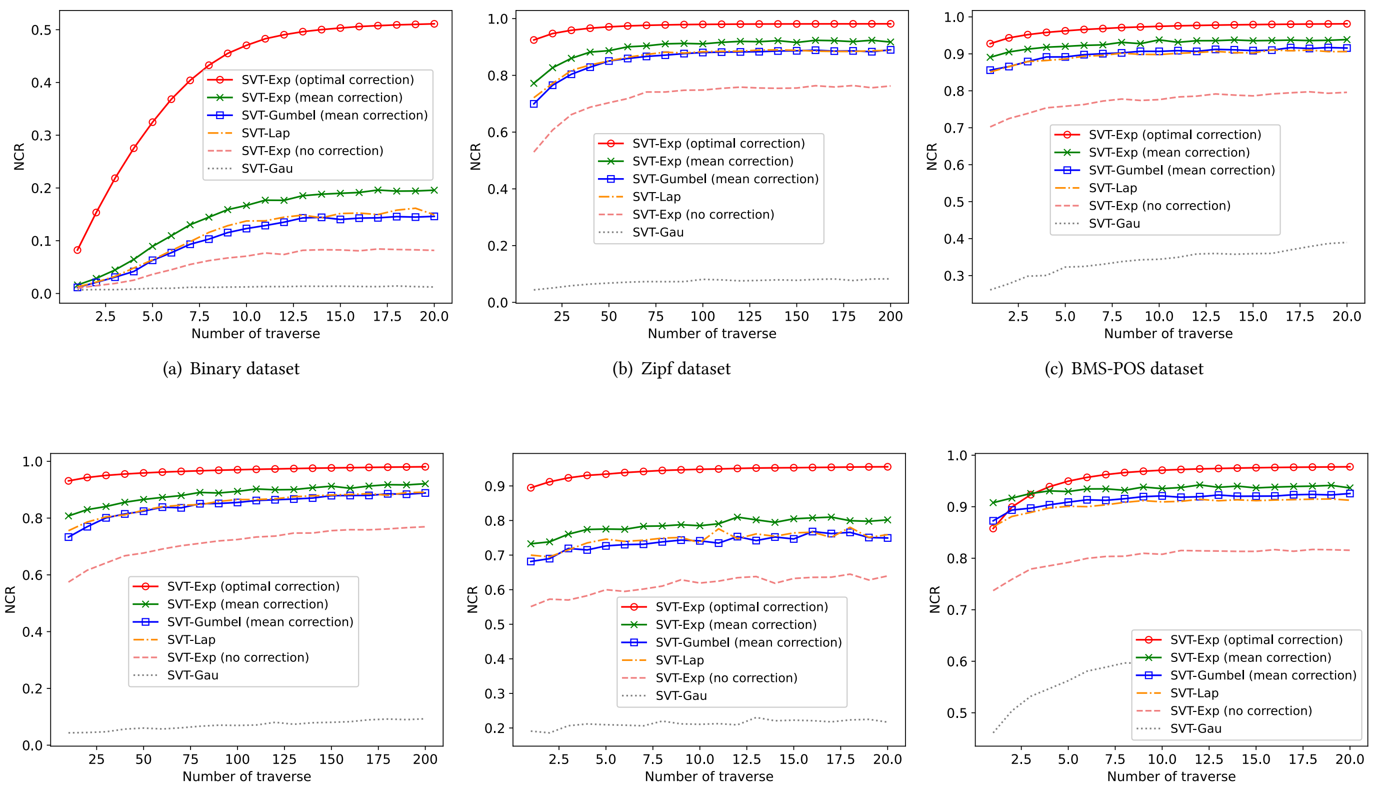
Section 4.3.2. Even though, it is worth noting that our method outperforms the others on the T40I10D100K dataset with only a few more traverses (*e.g.*, less than 5). That is also to say, our proposed method yields better performance compared to other baselines with only a marginal additional computation cost.

6 RELATED WORK

Our work focuses on enhancing one of the most fundamental DP algorithms initially introduced by Dwork et al. [14, 15, 38], namely the sparse vector technique (SVT). Benefiting from its key feature where only positive outcomes consume privacy, SVT and its variants [25] have become crucial components in numerous algorithms across various domains and scenarios [4, 7, 22, 24, 26, 41, 42, 47]. The applications of SVT span shared-parameter selection in deep learning models [41], time-stamp selection for streaming data [22], online query answering [4], among others. By improving the accuracy of SVT in a broad context, our work has the potential to enhance the performance of the aforementioned algorithms.

To gain deeper insights into the sparse vector technique, Lyu *et al.* summarize all prevalent SVT variants in [31]. Apart from the thorough privacy analysis under the classic notion of DP, they also propose an enhanced SVT with the optimal privacy budget allocation scheme. Shortly after, Zhu *et al.* [49] revisit the privacy analysis of the SVT algorithm under a relaxed privacy notion, namely Rényi differential privacy (RDP) [35]. Their research explores the utility of Gaussian noise for both threshold and query perturbation within the SVT framework. They argue that Gaussian noise may outperform Laplace noise in specific scenarios, such as when query results predominantly fall below predefined thresholds or when the number of queries is limited and not excessively large. In alignment with Lyu *et al.* [31], our work conducts the privacy analysis through the lens of DP, aiming for a generic result. Moreover, our work focuses more on noise selection rather privacy allocation, which distinguishes us from Lyu *et al.*. Also, we demonstrate the advantage of leveraging the exponential noise in comparison to other alternatives, which further distinguishes us from Zhu [49].

Another line of research improves the SVT by exploiting information revealed in the algorithm for free. Concretely, Ding *et al.* [11] pinpoint that releasing the gap between the noisy thresholds and the noisy query results incurs no extra privacy costs. Hence, they use the free gap as guidance to design a better privacy budget allocation scheme. Kaplan *et al.*. [25] improve SVT by iteratively deleting



(a) Binary dataset (b) Zipf dataset (c) BMS-POS dataset

(d) Kosarak dataset (e) Adult dataset (f) T40I10D100K dataset

Figure 8: NCR on six datasets with varying number of traverses. Parameter $c = 5$ for Adult and $c = 50$ for the remaining datasets, $\alpha = 0$, $k = \lfloor \frac{m}{c} \rfloor$. `RESAMPLE=False`, and `APPEND=True`. Sequential composition theorem is adopted for computing ε.

elements that contribute to current positive outcomes, which enables a more refined privacy accountant. Since both methods can be applied on top of our proposed method, we leave them out for comparison in this work.

Notice that we are not the pioneer effort in utilizing the exponential noise for achieving differential privacy in the literature. Previous studies from Durfee *et al.* [12] and Shekelyan *et al.* [40] demonstrate the use of exponential in achieving the exponential mechanism (EM), another differentially private algorithm primarily employed for privacy-preserving top-k selection tasks. Concretely, they [12, 40] leverage the exponential noise to Oneshot [37] mechanism where items with the top-k highest score are selected by injecting noise into each score ranking all noisy scores in descending order [10, 32, 45]. While the exponential mechanism (EM) offers superior accuracy guarantees compared to SVT for the top-k selection problem, its application is limited: EM struggles with queries on streaming data, where the total number of queries might be infinite. Therefore, our focus remains on SVT in this work, given its broader applicability as a more generic algorithm. Ding *et al.* [11] briefly look into applying the exponential noise to SVT. However, their approach, which involves injecting exponential noise into the threshold, has been demonstrated to compromise privacy by our findings. To the best of our knowledge, we are the first attempt that applies the exponential noise in SVT, accompanied by rigorous privacy and utility guarantee.

7 CONCLUSION

This work aims to enhance the query accuracy of the SVT algorithm by utilizing exponential noise. We revisit the privacy analysis of SVT algorithms and expanding the range of noise options for query perturbation by considering its less informative nature. Our analysis identifies exponential noise as the most effective one, both theoretically and empirically, among the considered noise distributions. Additionally, we develop a generic optimal threshold correction method and an appending strategy to ensure both a high query precision and recall for SVT with marginal computation cost. The effectiveness of these methods is thoroughly validated through comprehensive experiments on both real-world and synthetic datasets.

8 ACKNOWLEDGEMENT

This work is supported by National Key R&D Program of China (2023YFB4503600), by National Natural Science Foundation of China (U23A20299, 62072460, 62172424, 62276270, 62322214), and by Alibaba Group through Alibaba Research Intern Program.

REFERENCES

[1] Charu C Aggarwal, Yan Li, Jianyong Wang, and Jing Wang. 2009. Frequent pattern mining with uncertain data. In *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*. 29–38.

[2] Raef Bassily, Om Thakkar, and Abhradeep Guha Thakurta. 2018. Model-agnostic private learning. *Advances in Neural Information Processing Systems* 31 (2018).

[3] Barry Becker and Ronny Kohavi. 1996. Adult. UCI Machine Learning Repository. DOI: https://doi.org/10.24432/C5XW20.

[4] Mark Bun, Thomas Steinke, and Jonathan Ullman. 2017. Make up your mind: The price of online queries in differential privacy. In *Proceedings of the twenty-eighth annual ACM-SIAM symposium on discrete algorithms*. SIAM, 1306–1325.

[5] Craig Cahillane. 2024. Sum of Exponential and Laplace Distributions. https://ccahilla.github.io/sum_exponential_and_laplace_distributions.pdf

[6] Ricardo Silva Carvalho, Ke Wang, Lovedeep Gondara, and Chunyan Miao. 2020. Differentially private top-k selection via stability on unknown domain. In *Conference on Uncertainty in Artificial Intelligence*. PMLR, 1109–1118.

[7] Rui Chen, Qian Xiao, Yu Zhang, and Jianliang Xu. 2015. Differentially private high-dimensional data publication via sampling-based inference. In *Proceedings of the 21th ACM SIGKDD international conference on knowledge discovery and data mining*. 129–138.

[8] Graham Cormode, Somesh Jha, Tejas Kulkarni, Ninghui Li, Divesh Srivastava, and Tianhao Wang. 2018. Privacy at scale: Local differential privacy in practice. In *Proceedings of the 2018 International Conference on Management of Data*. 1655–1658.

[9] Bolin Ding, Janardhan Kulkarni, and Sergey Yekhanin. 2017. Collecting telemetry data privately. *Advances in Neural Information Processing Systems* 30 (2017).

[10] Zeyu Ding, Daniel Kifer, Thomas Steinke, Yuxin Wang, Yingtai Xiao, Danfeng Zhang, et al. 2021. The permute-and-flip mechanism is identical to report-noisy-max with exponential noise. *arXiv preprint arXiv:2105.07260* (2021).

[11] Zeyu Ding, Yuxin Wang, Yingtai Xiao, Guanhong Wang, Danfeng Zhang, and Daniel Kifer. 2023. Free gap estimates from the exponential mechanism, sparse vector, noisy max and related algorithms. *The VLDB Journal* 32, 1 (2023), 23–48.

[12] David Durfee and Ryan M Rogers. 2019. Practical differentially private top-k selection with pay-what-you-get composition. *Advances in Neural Information Processing Systems* 32 (2019).

[13] Cynthia Dwork. 2006. Differential privacy. In *International colloquium on automata, languages, and programming*. Springer, 1–12.

[14] Cynthia Dwork, Moni Naor, Omer Reingold, Guy N Rothblum, and Salil Vadhan. 2009. On the complexity of differentially private data release: efficient algorithms and hardness results. In *Proceedings of the forty-first annual ACM symposium on Theory of computing*. 381–390.

[15] Cynthia Dwork, Aaron Roth, et al. 2014. The algorithmic foundations of differential privacy. *Foundations and Trends® in Theoretical Computer Science* 9, 3–4 (2014), 211–407.

[16] Cynthia Dwork, Guy N Rothblum, and Salil Vadhan. 2010. Boosting and differential privacy. In *2010 IEEE 51st annual symposium on foundations of computer science*. IEEE, 51–60.

[17] Úlfar Erlingsson, Vasyl Pihur, and Aleksandra Korolova. 2014. Rappor: Randomized aggregatable privacy-preserving ordinal response. In *Proceedings of the 2014 ACM SIGSAC conference on computer and communications security*. 1054–1067.

[18] Giulia Fanti, Vasyl Pihur, and Úlfar Erlingsson. 2015. Building a RAPPOR with the unknown: Privacy-preserving learning of associations and data dictionaries. *arXiv preprint arXiv:1503.01214* (2015).

[19] Tom Fawcett. 2006. An introduction to ROC analysis. *Pattern recognition letters* 27, 8 (2006), 861–874.

[20] Quan Geng, Wei Ding, Ruiqi Guo, and Sanjiv Kumar. 2019. Optimal noise-adding mechanism in additive differential privacy. In *The 22nd International Conference on Artificial Intelligence and Statistics*. PMLR, 11–20.

[21] Sivakanth Gopi, Yin Tat Lee, and Lukas Wutschitz. 2021. Numerical composition of differential privacy. *Advances in Neural Information Processing Systems* 34 (2021), 11631–11642.

[22] Avinatan Hasidim, Haim Kaplan, Yishay Mansour, Yossi Matias, and Uri Stemmer. 2020. Adversarially robust streaming algorithms via differential privacy. *Advances in Neural Information Processing Systems* 33 (2020), 147–158.

[23] Peter Kairouz, Sewoong Oh, and Pramod Viswanath. 2015. The composition theorem for differential privacy. In *International conference on machine learning*. PMLR, 1376–1385.

[24] Haim Kaplan, Yishay Mansour, Shay Moran, Kobbi Nissim, and Uri Stemmer. 2023. On Differentially Private Online Predictions. *arXiv preprint arXiv:2302.14099* (2023).

[25] Haim Kaplan, Yishay Mansour, and Uri Stemmer. 2021. The sparse vector technique, revisited. In *Conference on Learning Theory*. PMLR, 2747–2776.

[26] Jaewoo Lee and Christopher W Clifton. 2014. Top-k frequent itemsets via differentially private fp-trees. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*. 931–940.

[27] Yuhan Liu, Sheng Wang, Yixuan Liu, Feifei Li, and Hong Chen. 2024. Unleash the Power of Ellipsis: Accuracy-enhanced Sparse Vector Technique with Exponential Noise. arXiv:2407.20068 [cs.CR] https://arxiv.org/abs/2407.20068

[28] Yuhan Liu, Tianhao Wang, Yixuan Liu, Hong Chen, and Cuiping Li. 2024. Edge-Protected Triangle Count Estimation under Relationship Local Differential Privacy. *IEEE Transactions on Knowledge and Data Engineering* (2024).

[29] Yuhan Liu, Suyun Zhao, Yixuan Liu, Dan Zhao, Hong Chen, and Cuiping Li. 2022. Collecting triangle counts with edge relationship local differential privacy. In *2022 IEEE 38th International Conference on Data Engineering (ICDE)*. IEEE, 2008–2020.

[30] Yixuan Liu, Suyun Zhao, Li Xiong, Yuhan Liu, and Hong Chen. 2023. Echo of Neighbors: Privacy Amplification for Personalized Private Federated Learning with Shuffle Model. *arXiv preprint arXiv:2304.05516* (2023).

[31] Min Lyu, Dong Su, and Ninghui Li. 2017. Understanding the Sparse Vector Technique for Differential Privacy. *Proceedings of the VLDB Endowment* 10, 6 (2017).

[32] Ryan McKenna and Daniel R Sheldon. 2020. Permute-and-Flip: A new mechanism for differentially private selection. *Advances in Neural Information Processing Systems* 33 (2020), 193–203.

[33] Sebastian Meiser and Esfandiar Mohammadi. 2018. Tight on budget? tight bounds for r-fold approximate differential privacy. In *Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security*. 247–264.

[34] Microsoft 2020. *Putting differential privacy into practice to use data responsibly*. Microsoft. https://blogs.microsoft.com/ai-for-business/differential-privacy/.

[35] Ilya Mironov. 2017. Rényi differential privacy. In *2017 IEEE 30th computer security foundations symposium (CSF)*. IEEE, 263–275.

[36] Henri J Nussbaumer and Henri J Nussbaumer. 1982. *The fast Fourier transform*. Springer.

[37] Gang Qiao, Weijie Su, and Li Zhang. 2021. Oneshot differentially private top-k selection. In *International Conference on Machine Learning*. PMLR, 8672–8681.

[38] Aaron Roth and Tim Roughgarden. 2010. Interactive privacy via the median mechanism. In *Proceedings of the forty-second ACM symposium on Theory of computing*. 765–774.

[39] Omid Shakeri and Mir Pedram. 2012. QtyT40I10D100K. UCI Machine Learning Repository. DOI: https://doi.org/10.24432/C5360W.

[40] Michael Shekelyan and Grigorios Loukides. 2022. Differentially Private Top-k Selection via Canonical Lipschitz Mechanism. *arXiv preprint arXiv:2201.13376* (2022).

[41] Reza Shokri and Vitaly Shmatikov. 2015. Privacy-preserving deep learning. In *Proceedings of the 22nd ACM SIGSAC conference on computer and communications security*. 1310–1321.

[42] Ben Stoddard, Yan Chen, and Ashwin Machanavajjhala. 2014. Differentially private algorithms for empirical machine learning. *arXiv preprint arXiv:1411.5428* (2014).

[43] Jun Tang, Aleksandra Korolova, Xiaolong Bai, Xueqiang Wang, and Xiaofeng Wang. 2017. Privacy loss in apple's implementation of differential privacy on macos 10.12. *arXiv preprint arXiv:1709.02753* (2017).

[44] Tianhao Wang, Jeremiah Blocki, Ninghui Li, and Somesh Jha. 2017. Locally differentially private protocols for frequency estimation. In *26th USENIX Security Symposium (USENIX Security 17)*. 729–745.

[45] Larry Wasserman and Shuheng Zhou. 2010. A statistical framework for differential privacy. *J. Amer. Statist. Assoc.* 105, 489 (2010), 375–389.

[46] Qingqing Ye, Haibo Hu, Xiaofeng Meng, and Huadi Zheng. 2019. PrivKV: Key-value data collection with local differential privacy. In *2019 IEEE Symposium on Security and Privacy (SP)*. IEEE, 317–331.

[47] Huanyu Zhang, Ilya Mironov, and Meisam Hejazinia. 2021. Wide network learning with differential privacy. *arXiv preprint arXiv:2103.01294* (2021).

[48] Zijian Zheng, Ron Kohavi, and Llew Mason. 2001. Real world performance of association rule algorithms. In *Proceedings of the seventh ACM SIGKDD international conference on Knowledge discovery and data mining*. 401–406.

[49] Yuqing Zhu and Yu-Xiang Wang. 2020. Improving sparse vector technique with renyi differential privacy. *Advances in Neural Information Processing Systems* 33 (2020), 20249–20258.