Front. Comput. Sci., 2025, 19(9): 199611
https://doi.org/10.1007/s11704-024-40952-3

REVIEW ARTICLE

A survey of estimating number of distinct values

Jiajun LI, Runlin LEI, Zhewei WEI (✉)

Gaoling School of Artificial Intelligence, Renmin University of China, Beijing 100872, China

Abstract Estimating the Number of Distinct Values (NDVs) is a critical task in the fields of databases and data streams. Over time, various algorithms for estimating NDVs have been developed, each tailored to different requirements for time, I/O, and accuracy. These algorithms can be broadly categorized into two main types: sampling-based and sketch-based. Sampling-based NDV algorithms improve efficiency by sampling rather than accessing all items, often at the cost of reduced accuracy. In contrast, sketch-based NDV algorithms maintain a compact sketch using hashing to scan the entire dataset, typically offering higher accuracy but at the expense of increased I/O costs. When dealing with large-scale data, scanning the entire table may become infeasible. Thus, the challenge of efficiently and accurately estimating NDVs has persisted for decades. This paper provides a comprehensive review of the fundamental concepts, key techniques, and a comparative analysis of various NDV estimation algorithms. We first briefly examine traditional estimators in chronological order, followed by an in-depth discussion of the newer estimators developed over the past decade, highlighting the specific scenarios in which they are applicable. Furthermore, we illustrate how NDV estimation algorithms have been adapted to address the complexities of modern real-world data environments effectively. Despite significant progress in NDV estimation research, challenges remain in terms of theoretical scalability and practical application. This paper also explores potential future directions, including block sampling NDV estimation, learning-based NDV estimation, and their implications for database applications.

Keywords number of distinct values, database, large-scale

1 Introduction

The challenge of estimating the number of distinct values (NDV) has a long history, beginning in 1943 with its application in biology to estimate the number of different species in nature [1]. This early work laid the foundation for NDV estimation in statistics, where researchers aimed to determine the support set size for discrete data distributions [2]. For example, NDV estimation has been used to gauge a writer's vocabulary by counting distinct words in their corpus [3].

In database systems, accurate NDV estimation is critical for tasks such as query optimization [4–7]. Systems like PostgreSQL and Spark rely on precise NDV assessments to improve query cardinality estimation, leading to more efficient query execution. For instance, improvements in NDV accuracy can reduce query execution times by up to 50%, as demonstrated in [8], while inaccurate estimates can cause significant slowdowns [6].

Beyond databases, NDV estimation has also proven essential in fields like network security, where counting distinct IP addresses helps in detecting potential attacks [9–11]. In machine learning, a related challenge involves studying rare occurrences in training data, which approximates the probability of generating hallucinations in language models [12].

Due to the extensive application of NDV estimation, several surveys have been conducted across various domains [13,14]. However, these surveys do not fully reflect the trends in the latest research [13] and are limited to specific types of estimators. Harmouch et al. [14] thoroughly investigate sketch-based NDV estimation but overlook sampling-based approaches. While sketch-based methods are well-established in practical applications, particularly in databases, the growing complexity of modern database environments has increased the relevance of sampling-based methods. These advancements have led to the development of new NDV estimators that have not been comprehensively reviewed.

Previous surveys have primarily focused on accuracy as the sole criterion for model evaluation. However, different scenarios demand a broader range of metrics. In streaming models, where algorithms must process data in a single pass, processing speed and memory usage are critical. In distributed models, in addition to time efficiency, I/O and communication costs are also crucial considerations.

We present a comparison of the different estimations produced by these two categories in Fig. 1. We are able to further subdivide the NDV estimation according to the design characteristics and analyze it in depth in the following sections.

By surveying recent algorithms, this paper delves into the core concepts of NDV estimation and sheds light on the insights behind various estimators. The main contributions are summarized as follows:

Received September 3, 2024; accepted December 16, 2024

E-mail: zhewei@ruc.edu.cn

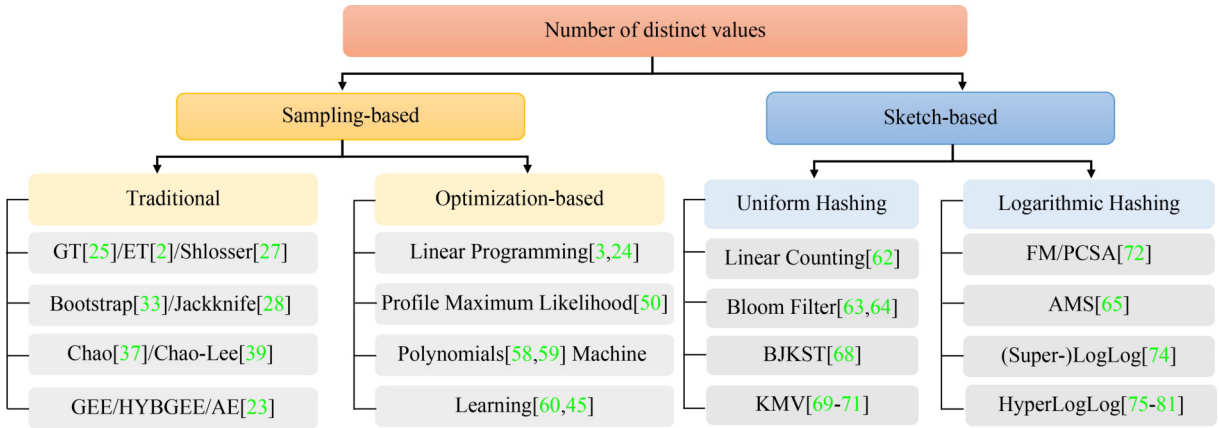


Fig. 1 The taxonomy of the number of distinct values estimation

- We comprehensively review the basic concepts and characteristics of NDV estimation and classify NDV estimation into two categories, sampling-based and sketch-based, based on the characteristics of the techniques. We systematically investigate the historical developments and key techniques.
- We present in detail the sampling-based and sketch-based algorithms and analyze in depth their advantages and disadvantages.
- We discuss the new problems and challenges that have arisen with NDV estimation in recent years as the database environment has changed, as well as the new algorithms and techniques that have arisen for NDV estimation to cope with the changing environments.

The rest of the paper is organized as follows. Section 2 provides the frequently used notations and background knowledge about NDV estimation. Next, we introduce the estimation of NDV from sampling-based and sketch-based approaches, corresponding to Section 3 and Section 4, respectively. Section 5 focuses on the adjustments and adaptions made to the NDV estimation based on the different data and environments in the real-world data. Section 6 surveys the important application, cardinality estimation of estimating NDV. We then discuss the possible future directions and the challenges in Section 7 and give the conclusion in Section 8.

2 Notation and background

This section provides a detailed definition of the problem associated with NDV. The frequently used notations are summarized in Table 1. In Fig. 1, we categorize NDV estimation methods into two types: sampling-based and sketch-based. We outline the key developments in each of these approaches. Before providing a detailed description of each estimator, we first introduce the necessary background. Initially, we introduce the fundamental notations and problem definitions. Subsequently, we discuss the frequently-used techniques frequently employed in sampling-based and sketch-based methods to avoid redundant explanations. Finally, we review the evaluation criteria relevant to this problem, which has led to a continuous updating of the research in different fields.

Table 1 Notations and the corresponding definitions

Notation	Description
N	Size of the population in the raw data
D	NDV of the raw data
q	Sample rate
n	Size of the population in the sample
n_j	Element j appear n_j times in the sample
N_j	Element j appear N_j times in the population
$\bar{N}$	Average class size in population, $\frac{N}{D}$
d	NDV of samples
$F_1,\ldots,F_i$	Number of items appear i times in population
$f_0, f_1,\ldots,f_i$	Number of items appear i times in all sample

2.1 Problem definition

In the context of database or data stream challenges, researchers are often faced with a collection of data. This data may manifest as a column within a database table or a series of IP addresses. Here, we define the concept of Frequency Vector.

Definition 1 (Frequency Vector / histogram)**.** We consider the multi-set $\boldsymbol{e} = \{e_1, e_2, \ldots, e_D\}$, where D denotes the number of distinct elements. Let N_j denote the appearing times of element e_j, then $N = \sum_{j=1}^{D} N_j$ is the population size of input data. For simplicity, we define source data $X = (N_1, N_2, ..., N_D)$. When we take a sample $S = \{e'_1, e'_2, ..., e'_n\}$, where n is the size of sample and e'_i is uniformly sampled from the input X. We denote n_j as the times of e_j appear in the sample, then we have $n = \sum_{j=1}^{d} n_j$, where d denotes the number of distinct values in the samples.

In order to compute arbitrary frequency moments of source data, we use the *frequency of frequency*, also known as profile and histogram of frequency. Its definition is as follows:

Definition 2 (Frequency of frequency)**.** The *frequency of frequency* in the raw data is represented by F_i, which is the number of elements that occur i times. For a given sampled subset, this frequency is signified by f_i, with f_i indicating the number of elements appearing i times in the sampled data. Notably, f_0 corresponds to the number of elements that are absent from the samples. Additionally, the *frequency ratio* for elements that occur i times is defined as $\frac{i}{N}$. Note that

frequence of frequency is also referred to as histograms of histograms [15], histogram order statistic [16], fingerprints [3] and profiles [17].

Example 1 (Frequency of frequency)**.** Given input data $E=\{1,1,2,3,3,4\}$, the frequency vector $X=(2,1,2,1)$ and the *frequency of frequency* of X is $\{2:2,1:2\}$. As a result, we have $F_1=2, F_2=2$. Using the *frequency of frequency* of X, the distinct values of X can be further calculated as $D(X)=2+2=4$. If we sample from X obtaining the subset $\{1,1,2\}$, then the frequency of frequency for the sampled subset is $f_0=2, f_1=1, f_2=1$.

The number of distinct values can be calculated from the *frequency of frequency*. This class of polynomial expressions for the *frequency of frequency* can be defined as frequency moments.

Definition 3 (kth order frequency moments)**.** Consider the multi-set $E=\{e_1,e_2,\ldots,e_D\}$ with the *frequency of frequency* F_i, where F_i denotes the number of elements appear i times. The kth order *frequency moments* $\mathcal{M}_k$ for each $k\geqslant 0$ are

$$\mathcal{M}_k=\sum_{i}^{D}F_i\cdot i^k.$$

The number of distinct values of E is the zeroth-frequency moment $\mathcal{M}_0$, which is the number of elements appearing at least once. We use D for $\mathcal{M}_0$ throughout the remainder of the paper. The number of population size N is exactly $\mathcal{M}_1$.

2.2 Common technologies

This subsection describes the common technologies applied in sampling-based and sketch-based NDV estimation. First, we introduce the concepts and techniques of these stochastic estimation algorithms. Then, we introduce the core techniques that are commonly used in sampling-based and sketch-based respectively.

Definition 4 ((ϵ,δ)-approximation scheme)**.** An (ϵ,δ)-approximation scheme for a quantity D is a (randomized) procedure that, given any positive $\varepsilon<1$ and $\delta<1$, computes an estimator $\hat{D}$ such that, with probability at least $1-\delta$, the estimator $\hat{D}$ is within a relative error ε. Formally, this means that $\Pr\left[\left|\hat{D}-D\right|\leqslant\varepsilon D\right]\geqslant 1-\delta$.

ϵ and δ are the parameters that control randomness and approximation. When there is a need to increase the probability of success, estimators or sketches often employ the median trick.

Definition 5 (Median trick)**.** Let $\mathcal{A}$ be a randomized algorithm that produces an estimator $\hat{D}$ within an error margin ϵ of the true value D with probability $p>0.5$. By running $\mathcal{A}$ independently $O(\log 1/\delta)$ times and taking the median of the resulting estimators, then $\Pr\left[\left|\hat{D}_{\text{median}}-D\right|\geqslant\varepsilon D\right]\leqslant\delta$.

The median trick is widely used in randomized algorithms to improve the probability of obtaining good estimates.

Block sampling Sampling-based NDV estimation utilizes samples taken uniformly from the source data. Traditional sampling-based algorithms take each item independently with equal probability, $\frac{1}{N}$. However, Brutlag et al. [18] point out that usually, in databases, data is stored in blocks on the hard disk. To achieve even sampling in a database, the process essentially requires scanning the entire table, which is no different from a full table scan. While scanning the entire table for sampling purposes can improve efficiency to some extent, it does not reduce the time complexity to sub-linear levels. The time complexity remains linear, as each data point needs to be considered to ensure an even and representative sample. Therefore, to realize sampling estimation of NDV, it is necessary to sample the data block uniformly. Uniformly sampling blocks of data can lead to data dependence in the samples, and there has been some work in recent years on how to use block sampling to estimate the NDV [18,19]. Ensuring that block sampling achieves both efficiency and accuracy simultaneously remains an unresolved challenge.

Similar to importance of sampling for sampling-based methods, the core technology of sketch-based method is hashing.

Hashing Sketch-based method typically feeds data into hashing functions to extract information about the data. Hashing functions are randomly picked from a family $\mathcal{H}$, which can map elements to range R. The independence of a hash function directly influences the number of collisions that occur between data items. We define the $t-$wise independent functions.

Definition 6 (t-wise independent function)**.** For every t distinct items $x_1,x_2,...,x_t$, and every output $y_1,y_2,...,y_t\in R$,

$$\Pr[h(x_1)=y_1\wedge h(x_2)=y_2\wedge\cdots\wedge h(x_t)=y_t]$$
$$=\prod_{i=1}^{t}\Pr[h(x_i)=y_i],$$

where h is chosen uniformly at random from $\mathcal{H}$.

Cryptographic hash functions, such as MD5, SHA-1, and SHA-3, are utilized when a highly independent hash function is required. However, in real applications, hash functions with fast operations, such as exclusive or bit shifts, are more popular. A popular example of hash functions in practice is murmurhash [1)], which is implemented by a series of fast bit manipulation operations and is very effective for applications.

2.3 Evaluation criteria

The criteria for evaluating errors are variable under different issues due to different application requirements. We describe the different evaluation criteria as follows. First, we define the relative error:

Definition 7 (Relative error)**.** The relative error of the estimation $\hat{D}$ is:

$$\mathcal{E}_{relative}(\hat{D})=|\hat{D}-D|/D.$$

[1)] See github.com/aappleby/smhasher website.

Some of the sketch-based estimators, such as Count-min sketch [20] and Bloom Filter [21], do not guarantee the relative error because they rely on estimators constructed from order statistics. Instead, they provide the additive error as follows:

Definition 8 (Additive error)**.** Give an estimation $\hat{D}$, the additive error of $\hat{D}$ is:

$$\mathcal{E}_{additive} = |D - \hat{D}|.$$

The additive error provides a measure of accuracy without regard to the magnitude of the true value, offering a straightforward, scale-independent metric of the estimation error.

In query optimization and cardinality estimation, it is common to use the Q-error [22] and ratio error [23] as the evaluation criterion, which is defined as:

Definition 9 (Q-error/ratio error)**.** The Q-error/ratio error of the estimation $\hat{D}$ is:

$$\mathcal{E}_{ratio}(\hat{D}) = \max\left\{\frac{\hat{D}}{D}, \frac{D}{\hat{D}}\right\}.$$

In statistical research, relative error is commonly used to assess the quality of an estimator. However, in the context of database research, factors such as time and space efficiency, along with their impact on data, are often more critical. As a result, metrics like Ratio error and Q-error are more frequently employed for database tasks. Additionally, when dealing with scenarios where the NDV is relatively small, additive error provides a more reliable error guarantee, helping to avoid extreme cases.

3 Sampling-based NDV estimation

This section reviews the development of the sampling-based estimators. We mainly categorize the sampling-based estimators into traditional and optimization-based estimators. Traditional estimators mainly refer to some methods before 2000 that answer the NDV estimation problem by designing a fixed, concise estimator. Optimization-based estimators, on the other hand, typically use optimization methods to adapt the different data inputs.

3.1 Traditional estimator

We illustrate the evolution of the traditional estimators through the introduction of four types of estimators.

Expectation-based classical estimators NDV estimation originates in 1943 [1] and the NDV estimator is first designed by Goodman [24] as follows:

$$\hat{D}_{Goodman} = d + \sum_{i=1}^{n}(-1)^{i+1}\frac{(N-n+i-1)!(n-i)!}{(N-n-1)!n!}f_i, \quad (1)$$

which is back-calculated from the expectation of f_i. $\hat{D}_{Goodman}$ is the unique unbiased estimator, when the sample size n is larger than $\max\{N_i\}$. When $n < \max\{N_i\}$, there is no unbiased estimator. In the earliest estimators, the practice of estimating frequent and infrequent elements separately was already considered. The distinction between frequent and infrequent elements has become a crucial approach in sampling-based NDV estimation.

Since unbiased estimators do not exist only when the sample rate n is too small and the coefficients of the f_i of $\hat{D}_{Goodman}$ do not converge, Good and Toulmin [25] consider a smooth estimator obtained by continuously increasing the number of samples of λn as follow:

$$\hat{D}_{GT} = d + \sum_{i=1}(-1)^{i+1}(\lambda-1)^i f_i, \quad (2)$$

where λ is an adjustable parameter that controls the sample rate. When the number of samples n is insufficient, increasing the number of samples to λn.

Efron and Thisted [2] utilize the Euler's transformation [26] to force oscillating series to converge rapidly and propose a new estimator, For some $\theta > 0$ and $J \in \mathbb{N}$,

$$\hat{D}_{ET} = d + \sum_{j=1}^{J}(-1)^{j+1}\theta^j b_j f_j, \quad (3)$$

where $b_j = \Pr[\text{Binomial}(J, 1/(\theta+1)) \geqslant j]$. J is also essentially an adjustable parameter that determines high and low frequencies.

Shlosser [27] also leverages the assumption that $\frac{E[f_i]}{E[f_1]} \approx \frac{F_i}{F_1}$, which holds when the population size N is large and the sampling fraction is nonnegligible. Shlosser [27] derives the estimator:

$$\hat{D}_{Shlosser} = d + \frac{f_1\sum_i(1-q)^i f_i}{\sum_i iq(1-q)^{i-1}f_i}. \quad (4)$$

According to [13], $\hat{D}_{Goodman}$ [24] and $\hat{D}_{Shlosser}$ [27] are the only two non-Bayesian estimators from sampling probability. They are similar in that the estimated error rate is large. According to [28], although Shlosser found that using $\hat{D}_{Shlosser}$ with a 10% sampling fraction resulted in an error rate below 20% in their experiment, the root mean squared errors still exceeding 200%. The effectiveness of the $\hat{D}_{Shlosser}$ is closely related to the data distribution. The estimator performs better with highly skewed data distributions, as these more effectively satisfy the underlying assumptions of the estimator.

Boostrap and jackknife estimator The main problem faced by NDV estimation is undersampling. To address this issue, a classical method named jackknife was proposed in 1949 [29]. The classic jackknife estimator constructs a new estimator by systematically omitting one observation at a time from the sample, producing $n-1$ subsamples, and then aggregating the estimates derived from these subsamples. For estimating the number of distinct species in a population, Burnham and Overton [30,31], Heltshe and Forrester [32] and Smith and van Bell [33] develop jackknife schemes. These jackknife estimators have a series of variants.

Another classic method to address undersampling is bootstrap, which is proposed by Bradley Efron in [34]. Bootstrapping resample from the samples or the distribution of samples to increase the number of samples for estimator design. Smith and van Bell [33] propose the bootstrapping

estimator as follows:

$$\hat{D}_{Boot} = d + \sum_{j:n_j>0} \left(1 - \frac{j}{n}\right)^n f_j. \tag{5}$$

$\hat{D}_{Boot}$ perform poorly when D is large and n is small. In [28], Hass and Stokes propose a generalized jackknife algorithm and discover the connections between the generalized jackknife approach and several other estimation approaches. Recall that d is the number of distinct values of samples. Hass et al. [28] derive the generalized jackknife estimator through the generalized jackknife estimator as follows:

$$\hat{D}_{JK} = d_n + K\frac{f_1}{n}, \tag{6}$$

where d_n denote the number of distinct values d from n samples to emphasize the dependence and K is an unknown real number. According to [28], the optimal value of K is:

$$K = \frac{D - \mathbb{E}[d_n]}{\mathbb{E}[f_1]/n}. \tag{7}$$

Following the settings of [27], if the population size N is large and the sampling fraction $q = n/N$ is nonnegligible, we have the below approximation:

$$\mathbb{E}[f_i] \approx \sum_{N_j} \binom{i}{N_j} q^i (1-q)^{N_j - i}. \tag{8}$$

If we assume that $\frac{\mathbb{E}[f_i]}{\mathbb{E}[f_i]} \approx \frac{F_j}{F_1}$ and $\mathbb{E}[f_0] = D - \mathbb{E}[d_n]$, the estimator is equivalent to $\hat{D}_{Shlosser}$, revealing the underlying connection between the estimators.

Hass et al. [28] discuss a number of estimators that can be obtained based on different approximations for K and $\mathbb{E}[f_1]/n$. With approximate each N_j in Eq. (8) by $\bar{N} = \frac{1}{D}\sum_{j=1}^{D} N_j = N/D$ and using Eqs. (6) and (7), they obtain the expression:

$$\hat{D} = d + \frac{(1-q)f_1\hat{D}}{n}. \tag{9}$$

By solving Eq. (9), they first derive the unsmoothed first-order jackknife estimator:

$$\hat{D}_{uj1} = (1 - (1-q)f_1/n)^{-1} d. \tag{10}$$

By approximate f_1 in Eq. (9) by $E[f_1] \approx n(1-q)^{\tilde{N}-1}$, they lead to the relation,

$$\hat{D} = d + \hat{D}(1-q)^{N/\hat{D}}. \tag{11}$$

$\hat{D}_{sj1}$ can be viewed as a solution of Eq. (11). Hass et al. [28] borrow the design of $\hat{D}_{CL}$ and derive the unsmoothed second-order jackknife estimator:

$$\hat{D}_{uj2} = \frac{d - f_1(1-q)\ln(1-q)\hat{\gamma}^2(\hat{D}_{uj1})/q}{1 - f_1(1-q)/n}, \tag{12}$$

where $\hat{\gamma}^2$ is an estimator of coefficient of variation (CV), γ^2, which is defined as:

$$\hat{\gamma}^2(D) = \max\left\{0, \frac{\hat{D}}{n^2}\sum_{i=1}^{n} i(i-1)f_i + \frac{\hat{D}}{N} - 1\right\}, \tag{13}$$

and the smoothed second-order jackknife estimator,

$$\hat{D}_{sj2} = \frac{d - (1-q)^{\tilde{N}}\ln(1-q)N\hat{\gamma}^2(\hat{D}_{uj1})}{1 - (1-q)^{\tilde{N}}}, \tag{14}$$

where $\tilde{N} = N/\hat{D}_{uj1}$. $\hat{D}_{uj1}, \hat{D}_{uj2}, \hat{D}_{sj1}$ and $\hat{D}_{sj2}$ are consistent for D.

Hass et al. [6] also propose a more complex version of the smoothed jackknife estimator

$$\hat{D}_{sjack} = \frac{d + Nh_n(\tilde{N})g_{n-1}(\tilde{N})\hat{\gamma}^2(\hat{D}_0)}{1 - (N - \tilde{N} - n + 1)f_1/(nN)}, \tag{15}$$

where h_n can be calculated by the standard gamma function Γ:

$$h_n(x) = \frac{\Gamma(N-x+1)\Gamma(N-n+1)}{\Gamma(N-n-x+1)\Gamma(N+1)},$$

for $x > 0$, and $\hat{\gamma}^2$ can be given by Eq. (12) and g_{n-1} is defined by

$$g_n(x) = \sum_{t=1}^{n} \frac{1}{N-x-n+t},$$

and $\hat{D}_0$ is defined by

$$\hat{D}_0 = (d - f_1/n)\left(1 - \frac{(N-n+1)f_1}{nN}\right)^{-1},$$

and $\tilde{N} = N/\hat{D}_0$.

Another frequently mentioned jackknife estimator is the Horvitz-Thompson estimator [35,36]. Based on the jackknife construction technique, the Shlosser estimator also has a corresponding iterative improvement scheme [28]. We can refer to [6,28] for the different versions of Horvitz-Thompson and Shlosser estimators for NDV estimation and their derivation process. According to the experiments of [28], the introduction of jackknife compensates for the errors due to under-sampling. However, since these jackknife estimators have complex expressions, they are not used much in practice. Despite its shortcomings, the jackknife estimator has greatly inspired subsequent ideas for integrating estimators for different data distributions.

Chao and Chao-Lee's estimator Although the origins of NDV research lie in counting the number of unseen species, its intensive study is closely linked to the development of the databases. The first estimator to be applied to databases is Chao's estimator [5].

Chao [37] first considers the relationship between the frequency moments. By assuming that N tends to infinity, Chao obtains the estimator:

$$\hat{D}_{Chao} = d + \frac{f_1^2}{2f_2}. \tag{16}$$

When $f_2 = 0$, $\hat{D}_{Chao}$ is ill-defined. According to [38], a bias-corrected estimator for Chao's Estimator is as follows:

$$\hat{D}_{Chao2} = d + \frac{f_1(f_1-1)}{2(f_2+1)}. \tag{17}$$

Due to the elegant design and outstanding effects, $\hat{D}_{Chao}$ can be easily embedded into programs. The first is an attempt by

Ozsoyoglu et al. [5] to integrate the estimator into the SQL statement.

After $\hat{D}_{Chao}$, Chao and Lee [39] design new estimators to cope with data from infinite distributions through the perspective of sample coverage. Sample coverage C is defined as the fraction of classes in the population that appear in the sample:

$$C = \sum_{j:n_j>0} \frac{N_j}{N}.$$

According to Turing et al. [40], $\hat{C} = 1 - f_1/n$ is used for sample coverage. To deal with the skew in the data, Chao and Lee [39] combine this coverage estimator with a correction term and obtain the estimator:

$$\hat{D}_{CL} = \frac{d}{\hat{C}} + \frac{n(1-\hat{C})}{\hat{C}}\hat{\gamma}^2, \tag{18}$$

where $\hat{\gamma}^2$ is also the estimator of coefficient of variation (CV) like Eq. (13), which is defined as:

$$\gamma^2 = \frac{(1/D)\sum_{j=1}^{D}(N_j - \bar{N})^2}{\bar{N}^2}.$$

In [39], an improved estimator of γ^2 is proposed as:

$$\hat{\gamma^2} = \max\left\{\frac{\hat{D}_1 \sum i(i-1)f_i}{n^2 - n - 1}, 0\right\},$$

where $\hat{D}_1$ is an initial estimator $\hat{D}_1 = d/\hat{C}$. With the different bias-corrected version of $\hat{\gamma}^2$, Chao et al. [39] propose the more complex version of $\hat{D}_{CL}$. The coefficient of variation γ is essentially a parameter that characterizes the distribution of data. Incorporating such a parameter into the estimator enables it to adapt to variations across different datasets more effectively. Though the $\hat{D}_{CL}$ family of estimators are not widely used as $\hat{D}_{Chao}$, the introduction of γ greatly influences the design of subsequent complex classical estimators, which we introduce next.

Integrated estimator The NDV estimation process is characterized by two primary features: the distinction between high and low-frequency elements and the differentiation between uniform and skewed distributions. These two aspects must be carefully integrated into the design of an effective estimator.

A significant trend in NDV estimation research is the distinction between high and low frequencies of occurrence. For instance, in the $\hat{D}_{Chao}$ estimator, the term $\frac{f_1^2}{2f_2}$ aggregates low-frequency information, while the parameter d consolidates high-frequency data. Additionally, it is crucial to differentiate between uniform and skewed distributions. In the case of Jackknife estimators, the derivation necessitates replacing N_j with $\bar{N}$ under the assumption that the data is evenly distributed. Conversely, the $\hat{D}_{Shlosser}$ estimator requires the condition $\frac{\mathbb{E}[f_i]}{\mathbb{E}[f_1]} \approx \frac{F_i}{F_1}$, which assumes a skewed distribution. As a result, these estimators are optimized for the specific types of distributions for which they are designed. The challenge in estimator design lies in effectively balancing the need to account for both frequency variations and distributional assumptions, ensuring accurate and robust estimation.

Charikar et al. [23] focus on observing extremes in the data distribution and using targeted estimators for different data distributions. Charikar et al. [23] also propose the hard case about distinguishing the following two situations:

- N same elements.
- $(N-t)$ same elements and t distinct elements.

In the second scenario, d is a good estimator only when the t distinct elements are all sampled. Based on the hard case, for any sampling-based estimator, with probability at least $\delta > e^{-n}$, the ratio error has the below lower bound:

$$\mathcal{E}_{ratio}(\hat{D}) \geqslant \sqrt{\frac{N-n}{2n}\ln\frac{1}{\gamma}}. \tag{19}$$

Charikar et al. [23] also propose a concise but efficient estimator for this extreme case as follows:

$$\hat{D}_{GEE} = d + \left(\sqrt{N/n} - 1\right) f_1. \tag{20}$$

Such extreme cases in datasets are rare in real-world applications, and for tasks like query optimization, exact NDV precision is often unnecessary. Minor inaccuracies typically have little impact on performance. In order to solve the case of different distributions, the first hybrid estimator is proposed by [6] as follows:

$$\hat{D}_{hybrid} = \begin{cases} \hat{D}_{sjack} & \text{if } u \leqslant x_{n-1,\alpha}, \\ \hat{D}_{Shlosser} & \text{if } u > x_{n-1,\alpha}, \end{cases} \tag{21}$$

where $u = \sum_{j:n_j>0} \frac{(n_j - \bar{n})^2}{\bar{n}}$ and $\bar{n} = n/d$. $x_{n-1,\alpha}$ is the α percentile of the Chi-square distributed random variable with $n-1$ degrees of freedom, where [6] sets $\alpha = 0.975$.

Charikar et al. [23] first replace the Shlosser's estimator for the skew part with $\hat{D}_{GEE}$ to obtain the hybrid estimator $\hat{D}_{HYBGEE}$. Considering that $\hat{D}_{HYBGEE}$ can not provide a theoretical guarantee like $\hat{D}_{GEE}$ for low-skew data, Charikar et al. [23] derive an adaptive estimator:

$$\hat{D}_{AE} = d + m - f_1 - f_2, \tag{22}$$

where m is the solution to the following equation:

$$m - f_1 - f_2 = f_1 \frac{\sum_{i=3} e^{-i} f_i + m e^{-(f_1+2f_2)/m}}{\sum_{i=3} i e^{-i} f_i + (f_1 + 2f_2) e^{-(f_1+2f_2)/m}}.$$

Higher-order expansions can yield more complex adaptive estimators, but, as with the $\hat{D}_{Goodman}$ estimator, this often leads to larger errors, making such complexity generally unnecessary. The process of solving for $\hat{D}_{AE}$ also sets the tone for subsequent estimator development. In practical NDV estimation, the estimator's ability to adapt to varying data distributions becomes crucial. A fixed estimator design often struggles to accommodate the diversity in input data effectively.

After 2000, there were fewer designs for traditional fixed-form estimators. This shift occurred because the worst-case scenarios presented in [23] posed significant challenges that are difficult to address using sampling-based algorithms.

Consequently, researchers begin to focus on developing estimators tailored to specific data distributions, as seen in studies such as [41–43], to avoid the limitations faced by traditional estimators.

3.2 Optimization-based estimator

In this subsection, we introduce sampling-based NDV estimation algorithms that utilize optimization techniques to adapt to various data inputs. These algorithms focus on determining the optimal number of samples needed to meet accuracy requirements. However, as noted in [23], there exist extreme cases where sampling-based methods may struggle to perform effectively. Nevertheless, modern algorithms prioritize real-world data scenarios over rare, challenging cases, focusing instead on exploring optimal sampling strategies.

Linear programming Solving NDV estimation with linear programming is a classic approach. As far back as Goodman's [24] original research, the relationship between F_i and f_i was found. In the past, constrained by limited computational resources, estimators had to be designed with simplicity in mind to minimize the number of computations required. However, with the advancement of computing power and the improved ability to solve linear programming problems, it has become feasible to solve NDV estimation using dynamic optimization techniques. Valiant et al. [3] first use multivariate central limit theorems (CLTs) and Stein's method to prove that obtaining support size within an additive constant error needs a lower bound of $\Omega(k/\log k)$, where k is the upper bound of support size. Studying the NDV estimation from a statistical point of view requires setting a probability that an element can be sampled at a minimum. When the data is confined to a finite dataset, the support size estimation problem can be viewed as equivalent to the NDV estimation problem. In this context, the upper bound of the support size, denoted as k, should be set equal to the total number of elements N in the dataset. Valiant and Valiant [3,44] are beginning with the relationship of $\mathbb{E}[f_i]$ and F_i.

The core idea is first to generate predetermined points $x_1,\ldots,x_\ell$ of $\{F'_{x_\ell}\}$. Notice that if $\{F'_{x_\ell}\}$ covers the original $\{F_i\}$, then we can derive the below relationship:

$$\mathbb{E}[f_i] = \sum_{j=1}^{\ell} F'_{x_j} \cdot poi(x_j, i), \tag{23}$$

where $poi(x_j, i)$ is the probability that it will be draw exactly i times in n independent draws from the element occurring x_j times using the Poissonization technique. Using the probabilistic relationship between F_j and f_i, along with the predetermined points j of F_j and input f_i, we can formulate an optimization problem to estimate the values of F_j. By solving this optimization problem, we can derive approximations for various properties [45], including NDV and entropy. The problem can be mathematically defined as follows: Given the *frequency of frequency* of samples $f_1, f_2, \ldots, f_m$ with the sample size n, define the vector $x = x_1, x_2, \ldots, x_\ell$ consisting of a fine mesh of points in the interval $(0, k]$. The objective is to solve the following linear program with variables $F'_{x_1},\ldots,F'_{x_\ell}$:

$$\begin{aligned} \text{Minimize:} \quad & \textstyle\sum_{i=1} \frac{1}{\sqrt{1+f_i}} \left| f_i - \sum_{j=1}^{\ell} F'_{x_j} \cdot poi(x_j, i) \right| \\ \text{Subject to:} \quad & \textstyle\sum_{j=1}^{\ell} x_j F'_j = \sum_{i=1} f_i, \text{and} \forall j, F'_j \geqslant 0. \end{aligned}$$

Valiant et al. do a series of research [3,44,46–48] based on the linear program from 2011 to 2017. This work fully discusses how to use linear programming to estimate support-related problems by adjusting the x_j to generate different $\{F_{x_j}\}$ [46], designing objective function [44,48] and application scenario [44,49]. The linear program algorithms offer the advantage of adapting to different datasets and providing a theoretically sound approach for solving the NDV estimation. However, they face several drawbacks, including inefficiency in solving the linear programs and instability in the estimated results, which make them difficult to apply in practice. Consequently, subsequent methods have built upon the concept of transforming NDV estimation into an optimization problem while avoiding the computational complexity associated with solving high-dimensional linear programs.

Profile maximum likelihood (PML) Another powerful tool for optimization is the maximum likelihood principle. Motivated by the principle of maximum likelihood, Orlitsky et al. [17] try to maximize the probability of the observed profile for the Profile Maximum Likelihood (PML) distribution. PML is slightly different from the previous maximum likelihood methods, which is defined as sequence maximum likelihood according to [50]. Their difference centers on the definition of the basic distribution, which can be shown clearly by the example in [50].

Example 2 ([50])**.** Suppose $X^N = \{x\ y\ x\}$, the the SML distribution is $(2/3, 1/3)$. However, the distribution in Δ that maximizes the probability of the profile $\varphi(x\ y\ x) = 1, 2$ is $(1/2, 1/2)$. Another example illustrating the power of PML to predict new symbols is $X^N = \{a\ b\ a\ c\}$, with profile $\varphi(a\ b\ a\ c) = 1, 1, 2$. The SML distribution is $(1/2, 1/4, 1/4)$, but PML is a uniform distribution over five elements, namely $\{\{1,1,1,1\},\{1,1,2\},\{1,3\},\{2,2\},\{4\}\}$.

The objective of the profile maximum likelihood estimator is summarized in [51] to the following optimization problem. Given n samples with empirical distribution $\hat{p} = (\hat{p}_{x_1}, \hat{p}_{x_2}, \ldots)$, maximize the probability to observe $\hat{p}$ up to relabeling $\pi\hat{p} \triangleq (\hat{p}_{x_{\pi(1)}}, \hat{p}_{x_{\pi(2)}}, \ldots)$ of the empirical distribution, where π is some permutation. This amounts to computing the PML distribution p^*:

$$p^* = \arg\max_{p} \sum_{\pi} \exp\left(-nKL(\pi\hat{p} \| p)\right), \tag{24}$$

where the argmax is over all discrete distributions p and the sum is over all permutations π of distribution p, and $KL(\cdot\|\cdot)$ is the Kullback-Leibler divergence.

According to [50], in comparison to the linear programming estimator, PML-based methods, by virtue of the maximum likelihood principle, are capable of identifying the distribution that best explains the entire profile of the observed data. Additionally, PML-based methods reduce the optimal sampling complexity, as outlined by Valiant et al. [3].

The experiments in [51] show that PML can better fit the distribution of data than Linear Programming. However, the profile maximum likelihood estimator is computationally challenging to find, and the best-known algorithm requires exponential time in support size. Therefore, efficient approximate computation algorithms have become the focus of research on PLM-based methods. Acharya et al. [52] try to use algebraic computation for PML. [53,54] combine the EM and MCMC algorithms for better results on PML distribution. Vontobel et al. [55,56] utilize Bethe approximation to approximate PML. Pavlichin et al. [51] propose an efficient algorithm for approximate computation of the profile maximum likelihood (APML). Despite the relatively good theoretical guarantees of the PML approach, in practice, it is rarely applied to database and data streaming tasks due to its high computational costs.

Polynomial approximation The idea of utilizing polynomials for NDV estimation originated from [57]. Both linear programming and the APML algorithms techniques used for solving NDV estimation and entropy estimation can inform and enhance each other. Wu et al. [58] propose the optimal estimation of the unseen based on Chebyshev polynomials. Comparing the former methods which the minimal sample size to achieve an additive error of εk at least 0.1 is within universal constant factors of $\frac{k}{\varepsilon^2 \log k}$, Wu et al. [58] improve the state-of-the-art result to $\frac{k}{\log k} \log^2 \frac{1}{\varepsilon}$. Wu et al. [58] begin with the linear estimator:

$$\hat{D} = \sum_{j=1}^{L} g_L(j) f_j + \sum_{j>L} f_j,$$

where $g_L(j)$ is a function to be determined with respect to j. As early as [24], we know that there is a linear correlation between NDV and low-*frequency of frequency*. By defining $g_L(j)$ as:

$$g_L(j) = \begin{cases} \frac{a_j j!}{n^j} + 1, & j \leqslant L, \\ 1, & j > L, \end{cases} \tag{25}$$

where a_j can be obtained from the Chebyshev polynomial and the binomial expansion and $L \triangleq \lfloor c_0 \log k \rfloor$, c_0 is a preset constant. The core idea of the polynomial approximation remains to approximate the low-frequency elements. Chebyshev polynomial approximation [58] provides a theoretical partition, L, for distinguishing between high and low-frequency elements. Subsequent research has also focused on improving the optimization and approximation intervals like PML. Chien [59] goes a step further and proves that polynomial NDV estimation is a weighted polynomial approximation problem. Since estimating the NDV is a weighted polynomial approximation problem, the Chebyshev polynomial approximation is only optimal for the uniform case and cannot cope with all the data distributions.

The sampling-based polynomial NDV estimation method highlights the essential elements (infrequent elements) that need to be computed in the NDV estimation. It provides a deep understanding of the fundamental source of the required sample size for NDV estimation. Additionally, this method is advantageous due to its low computational complexity.

Machine learning-based estimator Building on the foundation of polynomial NDV estimation, recent advancements have introduced machine learning-based approaches to enhance accuracy further. Eden et al. [60] developed a machine learning-based predictor to improve the precision of polynomial approximations. Given the inherent challenges in reducing the sampling complexity of NDV estimation, the integration of machine learning becomes essential for handling diverse data distributions. These learned methods are inherently tied to real-world data and applications, prioritizing effectiveness in practical scenarios over addressing worst-case situations.

In a virtualized network, Cohen et al. [10] first use online machine learning to predict the NDV of visiting the networks. According to Fig. 2 of [10], there is a strong correlation between f_1, f_2 and NDV and as a result, the complex modeling of the subsequent work continues to be built on the low *frequency of frequency* as well.

Wu et al. [61] propose a good machine-learning-based estimator, $\hat{D}_{WD}$ for NDV estimation in the database settings. They address three main fundamental issues for learning estimators: training data, model structure and generalization ability. Wu et al. [61] generate the training data to cover the variant distributions. Finding a linear correlation between low-frequency f_i and NDV, Wu et al. [61] use the truncated f_i, d and extra parameters as inputs and a multi-layer linear neural network to learn a regression predictor. They also design a loss function that enhances the generalization ability of the model from a theoretical point of view. Wu et al. [61] also prove that the learning-based methods perform optimally on the real-world data distributions by experiments. In the end, in order to evaluate the usability of the learning estimator, $\hat{D}_{WD}$ is deployed the predictor into the warehouse with user-defined functions (UDFs).

Recently, Li et al. [45] combine the principles of polynomial estimators with machine learning by treating the weight component as a trainable neural network. This approach leads to a more effective estimator that not only capitalizes on the theoretically optimal sampling complexity of polynomial methods but also adapts to varying data distributions. Additionally, this method has been extended to tackle more complex and diverse property estimation challenges, underscoring the potential for future advancements in NDV estimation techniques.

3.3 Summary

We summarize and compare the key features of sampling-based estimators in Table 2. Since sampling-based estimators avoid accessing the overall data, they are more efficient when resources are constrained, but their accuracy can not be guaranteed. Traditional estimators have very elegant expressions that are interpretable and suitable for use on static problems with known distributions. Overall, with the development, researchers have focused more on improving the accuracy of sampling-based estimators. As a result, sampling-based estimators have evolved from static fixed expression

Table 2 Comparison of sampling-based estimators in the literature review

Estimator	Fixed Expression	Linear	Optimization	Using q	Distribution
$\hat{D}_{Goodman}$ [24]	√	√	×	√	×
$\hat{D}_{GT}$ [40]	√	√	×	×	×
$\hat{D}_{ET}$ [2]	√	√	×	×	×
$\hat{D}_{Shlosser}$ [27]	√	√	×	√	×
$\hat{D}_{Boot}$ [33]	√	√	×	×	×
$\hat{D}_{uj1}, \hat{D}_{uj2}$ [28]	√	√	×	√	×
$\hat{D}_{sj1}, \hat{D}_{sj2}$ [28]	×	√	×	√	×
$\hat{D}_{Chao}$ [38]	√	×	×	×	×
$\hat{D}_{CL}$ [39]	√	×	×	×	×
$\hat{D}_{GEE}$ [23]	√	√	×	√	×
$\hat{D}_{hybrid}$ [6]	√	×	×	√	√
$\hat{D}_{AE}$ [23]	√	×	×	√	√
$\hat{D}_{LP}$ [44]	×	√	√	×	√
$\hat{D}_{APML}$ [51]	×	×	√	×	√
$\hat{D}_{WY}$ [58]	√	√	√	×	×
$\hat{D}_{WD}$ [61]	×	×	√	√	√

Remarks: "Fixed Expression" refers to whether the estimator has a fixed expression. "Linear" refers to whether the estimator is a linear combination of f_i. "Optimization" refers to whether the estimator is solved using optimization algorithms. "Using q" refers to whether the estimator is solved with sample rate q. This usually determines whether the data distribution is finite or infinite, which varies across research areas. "Distribution" refers to whether the estimator can adapt to different distributions.

designs to optimization-based approaches to accommodate the dynamic effects on the estimator due to various data distributions. Optimization-based estimator methods are better at incorporating newly generated data for more complex problems but correspondingly lack efficiency, making it more important to consider the trade-off between accuracy and efficiency.

4 Sketch-based NDV estimation

This section reviews the development of sketch-based NDV estimators. The sketch-based NDV estimators are roughly classified into two categories by [14]: uniform hashing and logarithmic hash algorithms. The former mainly uses a uniform hash function to hash the entire dataset into an interval, and the latter mainly tracks the most uncommon element observed so far. Uniform hashing algorithms perform well in datasets with small NDV, and the logarithmic hash algorithm handles large-scale datasets better. Sketch-based algorithms utilize a hash function to merge different elements. They only need to access each element once, with a total time complexity of $O(N)$ and a small space complexity.

4.1 Uniform hashing

Uniform sketch-based NDV estimation algorithms map elements to a small interval with a uniform hash function and proportionality between the size of the interval and NDV for estimation.

Linear counting sketch Whang et al. [62] design the Linear Counting (LC), which uses a bitmap B of size b that is a combination of all zeros to record the mapped bits by the hash function h. LC visits the input element x and hash it B, i.e., $B[h(x)] = 1$. LC extrapolates the NDV by counting the number of bits that are not mapped, which is denoted by V_0. The expected probability of a bit not being sampled is $\exp(D/b)$. Therefore, the maximum likelihood estimator of LC is:

$$\hat{D}_{LC} = -b\ln(V_0/b). \tag{26}$$

The accuracy of LC is controlled by the load factor $t = D/b$. The standard error of $\hat{D}_{LC}$ is $\sqrt{\frac{e^t - t - 1}{t*D}}$. Whang et al. [62] shows that using $t \leqslant 12$ provides $\hat{D}_{LC}$ with 1% standard error.

We provide a straightforward example below.

Example 3 (LC)**.** Given a random sequence $X = \{78, 38, 54, 93, 55, 16, 1, 49, 86, 48\}$, we set the hashing function as $h(x) = x\%7$. Now, b is set to 7, and only index 4 has no mapped value, so V_0 is 1, $D_{LC} = -7\ln(1/7) \approx 13.6$, while the ground-truth value is 10.

Although LC achieves high accuracy, its primary drawback is the requirement to have an approximate knowledge of the range of D and to preset the size of the bit array accordingly.

Bloom filter Bloom filter is designed to recognize whether an element exists in the bitmap or not. The main error of LC comes from the collisions of the bitmap. Bloom Filter can reduce the collisions by using m independent hash functions. Denote the total number of bits as V in Bloom Filter. Similar to how LC utilizes the relationship between the number of bits equal to 0 and the probability of the unmapped, intuitively, Swamidass and Baldi [63] estimate D using the estimator:

$$\hat{D}_{BF1} = -\frac{b}{m}\ln\left(1 - \frac{V}{b}\right). \tag{27}$$

Papapetrou et al. [64] derive a new estimator by the maximum likelihood as follows:

$$\hat{D}_{BF2} = \frac{\ln(1 - V/b)}{m\ln(1 - 1/b)}. \tag{28}$$

Besides, BF can be applied to estimate the cardinality of the intersections [64]. BF is a relatively easy algorithm to extend, and we will discuss its extensions in the following section. Similar to BF, many sketches designed for other purposes can be applied to NDV estimation, such as AMS [65], Count-Min [66] and Coordinated Sampling [67].

BJKST sketch A series of methods try to provide strong guarantees about $(\varepsilon, \delta)-$approximation scheme. The (ε, δ)-approximation scheme NDV estimation algorithm is first proposed by [67], namely Coordinated Sampling. Bar-Yossef et al. [68] subsequently propose three improved algorithms on the top of Coordinated Sampling, in conjunction with AMS [65], the third of which achieves a near-optimal space, called BJKST algorithm, an acronym of authors last names. BJKST maintains a buffer B to store the initial elements at a level of $Z = 0$. Referring to [14], with the insertion of x, BJKST algorithm keep track of $g(x), \rho(h(x))$, where $g()$ is a uniform pairwise independent hash function, $h()$ is a pairwise independent universal hash function and $\rho(y)$ represents the positions of the least significant 1-bit in y. The hash function

h guarantees that the probability of $\rho(h(x)) \geqslant r$ is precisely $1/2^r$ for any $r \geqslant 0$ stated in [65]. In terms of hash functions, BJKST uses both uniform hashing and logarithmic hashing. When the buffer size is larger than a preset threshold c/ε, where $c = 576$ in [68], the level Z is increased by one and remove the elements in B with a level less than new Z. Hashing function g is used for the identities instead of the actual names. After visiting the dataset, BJKST can use the buffer size and the level to construct the estimator:

$$\hat{D}_{BJKST} = |B| \cdot 2^Z. \tag{29}$$

BJKST can provide an $(\varepsilon,\delta)-$approximation scheme of D, when using the median trick for $O(\log(1/\delta))$ round. Compared to Linear Counting and Bloom Filter, BJKST removes some elements to achieve better space complexity. This means that spreading the data evenly over the axes and retaining some data points on the edges is more critical for NDV estimation, which is used in the algorithm KMV to be described next.

KMV sketch According to BJKST, the order statistics can perform well in NDV estimation. A typical estimation is the *kth* minimum value order statistics observable (KMV) estimator. The KMV estimator is also inspired by the algorithmic ideas in BJKST [68]. Following BJKST [68], Beyer et al. [69] use the *kth* smallest hash value to estimate D. Assuming the elements will be mapped to a range $1,\ldots,R$ by the hashing function h and the *kth* smallest hash value is denoted by v_k, the KMV estimator is given by:

$$\hat{D}_{KMV} = (k-1) \cdot R/v_k. \tag{30}$$

Beyer et al. [69] give the relative error bound of $\sqrt{\frac{2}{\pi(k-2)}}$. The authors also use KMV to support the multi-set operations among the partition synopses. We give an example of the KMV sketch below.

Example 4 (KMV)**.** Given a random sequence $X = \{78,38,54,93,55,16,1,49,86,48\}$, we set the hashing function as $h(x) = x\%13+1$ and $k = 4$. As the sequence elements arrive, we end up maintaining the fourth smallest element, $v_k = 4$. Then we have $\hat{D}_{KMV} = (4-1) \cdot 13/4 \approx 9.75$.

The KMV estimation accuracy can also be improved by the median trick. Due to KMV's desirable scalability and small memory space, there is a lot of related and subsequent research work. Cohen [70] first proposes the version of KMV with $k = 1$. Dasgupta et al. [71] provide a generalized version of KMV in the form of the theta-sketch framework.

4.2 Logarithmic hashing

The advantage of the logarithmic hashing algorithm is that it can estimate the number of different elements on a large scale in logarithmic levels of space. Logarithmic hashing algorithms are mainly divided into trailing 1s and leading 0s. The trailing 1s algorithm uses the number of trailing 1s in the hashing bit pattern. The leading 0s algorithm is similar to the trailing 1s algorithm, but instead of maintaining the number of bits, it only records the maximum hashing value.

FM sketch Flajolet and Martin [72] first use the leading zeros in a single pass to estimate the number of distinct values. Flajolet and Martin map the elements to a binary string of length L over the range $[0,\ldots,L-1]$, the pattern $0^k1\ldots$ appears with probability $\frac{1}{2^{k+1}}$. FM algorithm uses m bitmaps with length B, initialized to all 0 and R_i denotes the number of trailing 1s of the ith bitmap. The estimator of the FM algorithm is given as follows:

$$\hat{D}_{FM} = \frac{2^{\bar{R}}}{\varphi} \text{ with } \bar{R} = \frac{1}{m}\sum_{i=1}^{m} R_i, \tag{31}$$

where $\varphi = 0.77351$ is a statistical correction factor. In [72], Flajolet and Martin combine multiple estimators with arithmetic averages. This procedure is called Probabilistic Counting with Stochastic Averaging, namely PCSA. Alon et al. [73] give a simple analysis for a similar algorithm, which is introduced next.

AMS sketch Alon et al. [65] first define the frequent moments statistics:

$$\mathcal{M}_\ell = \sum_i i^\ell F_\ell. \tag{32}$$

The frequent moment of order 0 corresponds to NDV, while the frequent moment of order 1 corresponds to the total size of elements. Alon et al. [65] devise algorithms to compute frequency moments of any integer order not less than 0. Different from the estimation of $F_i (i > 0)$, AMS designs the algorithm for estimation F_0 through leading 0s alone. AMS is similar to FM, with the difference being that leading 0s is set to R and then estimated using:

$$\hat{D}_{AMS} = 2^R. \tag{33}$$

AMS only keeps track of only the maximum observable value R. The authors provide the AMS's guarantees a ratio error of at most δ with a probability of at least $1-\frac{2}{\delta}$ for any $\delta > 2$. With the introduction of leading 0s applications, the sketches of the subsequent 'LogLog' series further reduce the space required to estimate the NDV significantly through different forms of aggregation.

(Super-)LogLog sketch After more than a decade, Durand and Flajolet [74] refine the FM into a more realistic version of the LogLog algorithm. The LogLog algorithm uses Probabilistic Counting with Stochastic Averaging, abbreviated as PCSA. The LogLog algorithm utilizes the first k bits as the sequential number of buckets to map the truncated hash values to different m buckets. In each bucket, LogLog stores the leadings 0s of the truncated hash, denoted M_i. The LogLog algorithm uses the arithmetic mean of M_i for estimation as follows,

$$\hat{D}_{LogLog} = \alpha_m m 2^{\frac{1}{m}\sum M_i}, \tag{34}$$

where $\alpha_m = e^{-\gamma}\sqrt{2}/2 = 0.39701$ (γ is Euler's constant) in practical when $m \geqslant 64$ and the LogLog guarantees a standard error $1.3/\sqrt{m}$. Utilizing the first k bits to divide the hash value improves the utilization of space. The LogLog family of algorithms is named for the fact that it only requires space about $O(\log\log N_{\max})$, where $N_{\max}$ is a priori upper bound on NDV.

In [74], Durand and Flajolet provide an improved version of the LogLog named SuperLogLog. The SuperLogLog algorithm makes two improvements to LogLog, thus eliminating the extremes in the randomization process. The two improvements are summarized as the truncation rule and restriction rule. The truncation rule is to retain only the $m_0 = \lfloor \theta_0 m \rfloor$ smallest values and discard the rest, and θ_0 is set to 0.7 in the original paper. The SuperLogLog estimator can be written as:

$$\hat{D}_{SuperLogLog} = m_0 \tilde{\alpha}_m 2^{\frac{1}{m_0}\sum^* M_i}, \tag{35}$$

where $\tilde{\alpha}_m$ is also the modified constant to ensure the estimator is unbiased, and $\sum^*$ denotes the truncated sum. With the application of the truncation rule, accuracy indeed increases with a standard error of $\frac{1.05}{\sqrt{m}}$. The restriction rule, on the other hand, states that each M_i can use only $\lceil \log_2 \lceil \log_2 (\frac{N_{\max}}{m}) + 3 \rceil \rceil$ bits, that is, 5 bits for maximum NDV $N_{\max}$ well over 10^8. SuperLogLog enhances the LogLog by eliminating the extreme algorithm. One way to eliminate the effect of extreme values is using different aggregation functions, which is the next HyperLogLog algorithm.

HyperLogLog(++) sketch Flajolet et al. [75] propose the HyperLogLog algorithm with the hypergeometric mean to replace the arithmetic mean of LogLog, which makes HyperLogLog a near-optimal NDV estimation algorithm. The HyperLogLog estimator with the normalized bias corrected harmonic mean is:

$$\hat{D}_{HyperLogLog} = \alpha_m m \frac{m}{\sum 1/2^{M_j}}, \tag{36}$$

where α_m is the bias correction factor and $\alpha_m = 0.7213/(1 + 1.079/m)$, when $m \geqslant 128$. HyperLogLog guarantees a standard error of $1.04/\sqrt{m}$. The author also makes two corrections to deal with the $\hat{D}_{HyperLogLog}$ values that are beyond the specified range. For the small range, HyperLogLog uses the LC (26) to resolve the nonlinear distortions. In the large range ($D > 2^{32}$), which is limited by the collisions of the 32-bit hash function. The HyperLogLog uses a correction to the estimator as follows:

$$\hat{D}_{HperLogLog*} = -2^{32} \log(1 - \hat{D}_{HyperLogLog}/2^{32}). \tag{37}$$

The authors argue that HyperLogLog is near optimal due to the standard error being near $1/sqrtm$, but there are still a lot of subsequent studies that try to enhance the effectiveness of HyperLogLog.

To handle the more practice situation in PowerDrill [76], Heule et al. [77] improve HyperLogLog to HyperLogLog++ for larger NDV. HyperLogLog++ first replaces the 32-bit hash function with 64-bit to cover the NDV beyond 10^9 with a low additional cost in memory. Using the 64-bit hash function can also reduce the collisions. Besides, the authors show the bias of HyperLogLog with a 64-bit hash function. To correct the bias, the authors use k-nearest neighbor interpolation to get the bias. The authors also use LC for the small NDV and suggest the criteria for when to use which estimator. Further, HyperLogLog++ compresses the sparse representation to reduce memory consumption. Relying on this series of independent improvements to HyperLogLog, HyperLogLog++ has enhanced efficiency and accuracy and works well in PowerDrill and other databases.

In practice, HyperLogLog++ is one of the most widely used sketches. However, in terms of theory, the optimal algorithm is proposed by Kane et al. [78], who requires $O(1/\varepsilon^2 + \log\log n)$ bits to return the $(1+\varepsilon)-$approximation of NDV. Besides, the algorithm can update in $O(1)$ time.

4.3 Summary

Sketch-based methods using uniform hashing are easier to understand and perform better when the NDV is relatively small. In contrast, sketch-based algorithms using logarithmic hashing are better when the NDV is very large. The common advantage of sketch-based methods is their accuracy and efficiency. Moreover, the sketch-based methods are mergeable, which is not the case with the sampling-based methods. As sketch-based NDV estimation has reached a theoretical and practical bottleneck, few superior methods have emerged in this domain. Subsequent research has focused on integrating different sketches or identifying gaps between existing sketch-based algorithms, as seen in works such as [79–81]. With the increasing magnitude of NDV and varying environmental factors, we next explore advancements in both sampling-based methods, which offer better scalability, and sketch-based methods, which are more user-friendly and efficient.

5 Extension of NDV estimation

In this section, we present some follow-up work related to NDV estimation in practice. In practice, environments are more complex and do not fully meet the idealized conditions assumed by many algorithms. As a result, numerous subsequent studies have emerged to address these challenges related to NDV estimation. First, we present some of the most recent, up-to-date research on NDV estimation for different environments, especially distributed environments. Secondly, we focus on the scenario that needs to be faced for sampling-based NDV in the more realistic database, i.e., how block sampling can be used to achieve NDV estimation. Finally, we review the development of the sketch-based NDV estimation on efficiency and effectiveness.

5.1 Distributed sampling-based NDV estimation

In general, the applicability of an algorithm in a distributed environment depends on whether the results produced by different machines running the algorithm can be effectively merged without incurring significant additional overhead. We should notice that most sketch-based algorithms are mergeable. Thus, mergeable sketch-based algorithms can quickly answer the NDV estimation in the distributed environment by simply merging the data sketches. While sketch-based methods are well-suited for handling distributed environments, these environments typically involve large volumes of data. Sketch-based methods require processing the entire dataset. However, when the dataset is particularly large, performing a full table scan results in substantial I/O costs, which may become prohibitively expensive. introduce a novel approach that integrates sampling-based and sketch-based

methods to achieve efficient NDV (Number of Distinct Values) estimation in distributed environments. Their method is designed to maintain low I/O costs while ensuring high accuracy. They address the challenge of calculating the "*frequency of frequency*" in skewed data distributions by transforming part of the problem into estimating the intersection size of multi-sets. Additionally, they adapt existing estimators, such as $\hat{D}_{Chao}$, $\hat{D}_{CL}$, and $\hat{D}_{Shlosser}$, to ensure compatibility with mergeable sketches, enabling accurate computation of the required features. This approach is complemented by the findings of Cormode et al. [82], who show that HyperLogLog can be used for multi-set intersection sizes. Instead of transmitting frequencies in distributed environments, Li et al. [83] collect the information by sketches and merge them at the endpoints to estimate the low *frequency of frequency*. They leverage Count Sketch [84] to estimate the second-order frequency moments, using these as features to address the requirements of $\hat{D}_{CL}$. The framework idea is easily capable of combining different sampling estimators for the distributed environment. Besides, it can be used for the estimation of other statistics as well, simply by combining machine learning and estimating features by sketches.

Wang et al. [85] similarly exploit the fact that HyperLogLog can utilize a small amount of space to compute the intersection and propose a dynamic sketch of NDV estimation that supports insertion and deletion. For another estimation issue, Wang et al. [86] use high-frequency vector and fast AGMS to answer the inner product question.

5.2 Block sampling-based NDV estimation

After reviewing the research on NDV estimation in distributed environments, we now turn our attention to the practical challenges of sampling-based NDV estimation, particularly in optimizing it for database environments. Sampling-based NDV estimation is typically employed in very large databases, which often cannot store all data in memory. In such cases, these databases usually compress and store data in blocks or pages, adding another layer of complexity to the estimation process. Unfortunately, according to [18], the cost of sampling the data uniformly at random is already the same as scanning the entire table. For the situation that uniformly sampling on records is not feasible, Brutlag et al. [18] choose block sampling as an alternative. They sample b blocks from the total B blocks. Different from using f_i, they define the *frequency of frequency* of block G_i and g_i. G_i denotes the number of distinct values occurring in exactly i blocks in the population. g_i denotes the number of distinct values occurring in exactly i blocks in the sample. With the redefinition for block sampling, they modify the previous estimator for estimation. A typically modified version is about $\hat{D}_{GEE}$ that defines the block sampling estimator as follows:

$$\hat{D}_{BGEE} = \sqrt{\frac{B}{b}} g_1 + \sum_{i \geqslant 2} g_i. \tag{38}$$

The experiments of [18] show that the proposed block sampling estimators are competitive with record sampling estimators. For the clustered data, the block sampling estimators consistently perform better than the record sampling estimators. In a subsequent work [19], this operation on block sampling is referred to as COLLAPSE. The COLLAPSE strategy involves counting identical elements within a block only once, regardless of their frequency in that block. As a result, the estimator proposed in [19] for the transformation described in [23] is equivalent to the expression found in [18]. Charikar et al. [23] give a negative result that a uniform-random sample of n tuples from a table of N tuples, no distinct-value estimator can guarantee a ratio error $< O(\sqrt{N/n})$. Chaudhuri et al. [19] show that with a block-level sampling of n tuples, this lower bound can be strengthened to $O(\sqrt{Nn_b/n})$, where n_b is the number of tuples per block. Chaudhuri et al. [19] also propose a two-phase adaptive block sampling algorithm. The two-phase algorithm calculates the sample size required for the desired accuracy and dynamically takes the additional samples in the second phase of sampling. Both [23] and [19] show that under block sampling, the traditional uniform estimator will have a relatively high loss of accuracy. Currently, **COLLAPSE** is a straightforward somewhat ad-hoc heuristic. With the development of machine learning, more methods are available to process sequence data [87], perhaps further improving the estimation of block sampling. Shekelyan et al. [87] propose the Hidden Shuffle method specifically for sequential data. Their empirical results demonstrated that this approach effectively reduces both memory consumption and sampling time.

From an engineering point of view, in a distributed environment, the data is pre-processed to transform into a random sample partition (RSP) [88,89]. Both Wei et al. [88], and Salloum et al. [89] demonstrate that the expected sample distribution of RSP data blocks is equivalent to that of the original large data table. However, RSP introduces double the space overhead and does not ensure that newly inserted data remains randomly distributed. The potential for further research into NDV and other statistical estimations using block sampling remains significant.

5.3 Innovative sketch-based NDV estimation

Enhancement of sketches Sketch-based NDV estimation can also leverage optimization techniques to enhance the functionality and performance of sketches. Debnath et al. [90] apply machine learning to improve the accuracy of set intersection cardinality of the Bloom filter, and Guo et al. [91] develop the dynamic bloom filters to better handle fluctuating data. These advancements underscore the effectiveness of optimization techniques in improving sketch-based algorithms. As the focus shifts to more advanced sketch-based algorithms, the discussion naturally leads to HyperLogLog, a method recognized for its status as one of the most classical and effective NDV estimators. The SQL **Approx_count_distinct** is supported in many database services to approximate the number of distinct elements, which is implemented using HyperLogLog [77]. Therefore, a series of works have centered around enhancing HyperLogLog, focusing on updating efficiency or improving effectiveness. Ertl [92] notices the deficiencies of the original estimator in small and large NDVs. Then, they designed the Maximum

Likelihood estimator to eliminate deviations using a numerical optimization algorithm. The estimator can be applied in joint NDV estimation to replace the inclusion-exclusion methods. The final version of the Maximum Likelihood estimator is designed as UltraLogLog [93], which can reduce the space by 24% while guaranteeing the same effect as HyperLogLog. UltraLogLog has been released as part of the open-source Hash4j library. The use of optimization algorithms, including traditional optimization algorithms and machine learning algorithms, to improve sketches has become an important direction for future research.

Integration and expansion of sketches Other sketch-based approaches focus on identifying the relationships between different sketch tasks and leveraging these connections to improve sketch performance. Tsan et al. [94] give a concrete implementation of how to design a learning-based approach to enhance classical sketches, Count-Min and Fast-AGMS. They present a method for converting a Count-Min sketch into a Fast-AGMS sketch. In order to optimize the sketch algorithm, introducing the advantages of other sketch algorithms is a common approach. Ertl [81] also explores the relationship between Minhash and HyperLogLog. Utilizing the link between sketches can synthesize the strengths and weaknesses of estimators for better results. Wang et al. [95] introduce a new class of estimators called τ-Generalized-Remaining-Area (τ-GRA) estimators, as well as a dramatically simpler approach to analyzing estimators. They also derive τ-GRA-based estimators for the class of Curtain sketches introduced by Pettie et al. [96], which can be seen as a hybrid of HyperLogLog and PCSA with a more attractive simplicity-accuracy tradeoff than both.

6 Cardinality estimation: the most important application of NDV estimation in practice

NDV estimation serves as the foundational basis for accurate cardinality and is a specific instance of cardinality estimation. Cardinality estimation is also the most important application of NDV estimation. The cardinality estimation generally requires two parts: the NDV of joins and selectivity. Systems like PostgreSQL (PG) and Spark will use NDV for the estimation of joins. As machine learning continues to advance, cardinality estimation has emerged with a wealth of new research opportunities. Given the critical role of cardinality estimation, which is a key application in AI4DB, extensive research has focused on estimating the NDV of joins using techniques such as sampling or sketching [97,98]. The accurate estimation of join size serves as the foundation of cardinality estimation, driving numerous efforts to explore NDV estimation in join operations.

The main learned cardinality estimators can be categorized into two types: Query-driven and Data-driven [99]. Query-driven estimators train on a set of labeled training queries to predict the cardinality of a given query, including [99–102]. Query-driven estimators encode queries as features to mitigate the impact of data dependencies. However, these methods require substantial amounts of data to train the model effectively, which significantly restricts their applicability. Additionally, query-driven approaches often rely on more complex models to further reduce the influence of data dependencies.

Compared to the query-driven methods, DeepDB [103] leads the way in the development of data-driven approaches. DeepDB [103] introduces the Sum-Product Network (SPN) to hierarchically decompose tables into simpler distributions. The hierarchical splitting ensures that partitions of different columns remain independent, while partitions of the same column can be aggregated via summation. A vertical split corresponds to a Product node in the SPN, where the join cardinality is the product of the individual partition cardinalities due to their independence. Conversely, a horizontal split corresponds to a Sum node. To balance space and accuracy, the number of partitions is controlled by a threshold. Another data-driven approach, NeuroCard [104], employs join sampling and modern deep autoregressive models but is generally more time-intensive.

Query-driven methods typically result in higher errors compared to data-driven methods, as they rely heavily on the specific training queries used during their development. However, query-driven methods have smaller inference/training time and model size and can handle extreme queries, such as cyclic join.

With the development of cardinality estimation, blending the advantages of query-driven and data-driven methods becomes the trend afterward [99]. Li et al. [8] propose an attention-based learned cardinality estimator, ALECE, designed for SPJ (Selection, Projection, Join) queries. ALECE explores and utilizes attention mechanisms to integrate the relationship between queries and dynamic data. ALECE is both data- and query-driven. Besides, ALECE performs well on dynamic workloads that mix queries and data manipulation statements, including inserts, deletes and updates. ALECE is integrated into the PostgreSQL system and outperforms its built-in optimizer. It is worth noting that the data feature chosen for ALECE is the histogram of elements, which is consistent with the NDV estimates. Applying the theory of NDV estimation to cardinality estimation can assist in designing theoretically guaranteed cardinality estimators.

7 Future trends and challenges

This section focuses on possible future extensions based on the current NDV estimation study.

7.1 Sampling-based property estimation

The problem of support size estimation, NDV estimation in a statistical context, in conjunction with polynomial theory, has attracted some attention in recent years. [58,60,105] After combining polynomials in depth, the problems of estimating the statistics on discrete distribution with *frequency of frequency* are unified into the problem of property estimation [106]. Given the increasing diversity and widespread application of property estimation, a promising direction for future research would be to explore how current NDV estimation techniques, including machine learning approaches, can be integrated into the study of property estimation.

Combining polynomials to do property estimation in the context of databases has proved to be an interesting problem [45], and more properties are waiting to be studied subsequently as well.

7.2 NDV Estimation in Complex Environments

Although there are mature methods for property estimation using sketch or sampling-based algorithms, these techniques face significant challenges in more complex environments. First, in distributed settings, reducing the communication cost of computing each *frequency of frequency* in a sampling-based approach remains difficult. While Li et al. [83] offers a method using the inclusion-exclusion principle to calculate the communication overhead for HyperLogLog intersection results, the communication cost of emerging learning-based HyperLogLog intersection methods still warrants further investigation. Second, when data is obtained through block sampling, leveraging the sequential information within these blocks to improve estimator performance is an open problem. In addition, Shan et al. [107] propose a novel data structure to handle the cardinality estimation of flows for massive high-speed traffic over sliding windows. Finally, in data stream environments, detecting distribution shifts and selecting appropriate estimators or retraining models in response to these shifts are critical areas for further research.

7.3 Non-linear learning-based NDV estimation

While there are now a number of approaches that utilize machine learning to enhance the effectiveness of NDV estimators [45,61,106], they all focus on the linear structure of NDV estimation. However, in practice, nonlinear estimators achieve superior results, such as $\hat{D}_{Chao}$ [45,61,83]. Considering that nonlinear models offer the potential to capture complex patterns and relationships that linear models might overlook, they are a promising avenue for future research.

In conclusion, while progress has been made, the field of NDV estimation still presents numerous challenges and opportunities. Further exploration of nonlinear models, along with addressing issues such as scalability and efficiency, could lead to significant breakthroughs in this area.

8 Conclusion

The number of distinct value estimation, a fundamental problem in databases, that can be applied to many extended tasks, has been studied for decades. In this paper, we conduct a comprehensive survey on the key techniques of NDV estimation, including sampling, sketch methods and the extension settings. We have summarized the contexts in which different NDV estimation methods are applicable and point out their shortcomings as well as directions in which they can be improved. NDV estimation and relative problem still face challenges for lager scale data. With the development of machine learning, more accurate and efficient NDV estimation can improve the downstream task in this field. We hope to provide valuable reference for researchers in this field and promote the further development of theories and methods related to NDV estimation. More future research will focus on estimators that utilize the *frequency of frequency* as features to estimate the properties of discrete data, such as frequency moments, entropy. With the optimization of these basic statistics, the database will be able to handle larger data, bringing help for data storage, query and analysis.

Acknowledgements This research was supported in part by the National Science and Technology Major Project (2022ZD0114802), the National Natural Science Foundation of China (Grant Nos. U2241212, 61932001), the Beijing Natural Science Foundation (No. 4222028), by the Beijing Outstanding Young Scientist Program (No.BJJWZYJH012019100020098), the Huawei-Renmin University joint program on Information Retrieval. We also wish to acknowledge the support provided by the fund for building world-class universities (disciplines) of Renmin University of China, by Engineering Research Center of Next-Generation Intelligent Search and Recommendation, Ministry of Education, by Intelligent Social Governance Interdisciplinary Platform, Major Innovation & Planning Interdisciplinary Platform for the "Double-First Class" Initiative, Public Policy and Decision-making Research Lab, and Public Computing Cloud, Renmin University of China.

Competing interests The authors declare that they have no competing interests or financial conflicts to disclose.

References

1. Fisher R A, Corbet A S, Williams C B. The relation between the number of species and the number of individuals in a random sample of an animal population. Journal of Animal Ecology, 1943, 12(1): 42−58
2. Efron B, Thisted R. Estimating the number of unseen species: how many words did Shakespeare know? Biometrika, 1976, 63(3): 435–447
3. Valiant G, Valiant P. Estimating the unseen: an n/log(n)-sample estimator for entropy and support size, shown optimal via new CLTs. In: Proceedings of the 43rd Annual ACM Symposium on Theory of Computing. 2011, 685–694
4. Hou W C, Ozsoyoglu G, Taneja B K. Processing aggregate relational queries with hard time constraints. In: Proceedings of 1989 ACM SIGMOD International Conference on Management of Data. 1989, 68–77
5. Ozsoyoglu G, Du K, Tjahjana A, Hou W C, Rowland D Y. On estimating count, sum, and average relational algebra queries. In: Proceedings of the International Conference on Database and Expert Systems Applications. 1991, 406–412
6. Haas P J, Naughton J F, Seshadri S, Stokes L. Sampling-based estimation of the number of distinct values of an attribute. In: Proceedings of the 21st International Conference on Very Large Data Bases. 1995, 311–322
7. Lemire D, Kaser O. Reordering columns for smaller indexes. Information Sciences, 2011, 181(12): 2550−2570
8. Li P, Wei W, Zhu R, Ding B, Zhou J, Lu H. ALECE: an attention-based learned cardinality estimator for SPJ queries on dynamic workloads. Proceedings of the VLDB Endowment, 2023, 17(2): 197−210
9. Chabchoub Y, Chiky R, Dogan B. How can sliding HyperLogLog and EWMA detect port scan attacks in IP traffic? EURASIP Journal on Information Security, 2014, 2014(1): 5
10. Cohen R, Nezri Y. Cardinality estimation in a virtualized network device using online machine learning. IEEE/ACM Transactions on Networking, 2019, 27(5): 2098−2110
11. Clemens V, Schulz L C, Gartner M, Hausheer D. DDoS detection in P4 using HYPERLOGLOG and COUNTMIN sketches. In: Proceedings of NOMS 2023-2023 IEEE/IFIP Network Operations and Management Symposium. 2023, 1–6
12. Kalai A T, Vempala S S. Calibrated language models must hallucinate.

In: Proceedings of the 56th Annual ACM Symposium on Theory of Computing. 2024, 160–171
13. Bunge J, Fitzpatrick M. Estimating the number of species: a review. Journal of the American Statistical Association, 1993, 88(421): 364−373
14. Harmouch H, Naumann F. Cardinality estimation: an experimental survey. Proceedings of the VLDB Endowment, 2017, 11(4): 499−512
15. Batu T, Fortnow L, Rubinfeld R, Smith W D, White P. Testing that distributions are close. In: Proceedings of the 41st Annual Symposium on Foundations of Computer Science. 2000, 259–269
16. Paninski L. Estimation of entropy and mutual information. Neural Computation, 2003, 15(6): 1191−1253
17. Orlitsky A, Santhanam N P, Viswanathan K, Zhang J. On modeling profiles instead of values. In: Proceedings of the 20th Conference on Uncertainty in Artificial Intelligence. 2004, 426–435
18. Brutlag J D, Richardson T S. A block sampling approach to distinct value estimation. Journal of Computational and Graphical Statistics, 2002, 11(2): 389−404
19. Chaudhuri S, Das G, Srivastava U. Effective use of block-level sampling in statistics estimation. In: Proceedings of 2004 ACM SIGMOD International Conference on Management of Data. 2004, 287–298
20. Cormode G, Muthukrishnan S. An improved data stream summary: the count-min sketch and its applications. Journal of Algorithms, 2005, 55(1): 58−75
21. Bloom B H. Space/time trade-offs in hash coding with allowable errors. Communications of the ACM, 1970, 13(7): 422−426
22. Li B, Lu Y, Wang C, Kandula S. Q-error bounds of random uniform sampling for cardinality estimation. 2021, arXiv preprint arXiv: 2108.02715
23. Charikar M, Chaudhuri S, Motwani R, Narasayya V. Towards estimation error guarantees for distinct values. In: Proceedings of the 19th ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems. 2000, 268–279
24. Goodman L A. On the estimation of the number of classes in a population. The Annals of Mathematical Statistics, 1949, 20(4): 572−579
25. Good I J, Toulmin G H. The number of new species, and the increase in population coverage, when a sample is increased. Biometrika, 1956, 43(1-2): 45−63
26. Bromwich T J I A. An Introduction to the Theory of Infinite Series. Providence: American Mathematical Society, 2005
27. Shlosser A. On estimation of the size of the dictionary of a long text on the basis of a sample. Engineering Cybernetics, 1981, 19(1): 97−102
28. Haas P J, Stokes L. Estimating the number of classes in a finite population. Journal of the American Statistical Association, 1998, 93(444): 1475−1487
29. Quenouille M H. Problems in plane sampling. The Annals of Mathematical Statistics, 1949, 20(3): 355−375
30. Burnham K P, Overton W S. Estimation of the size of a closed population when capture probabilities vary among animals. Biometrika, 1978, 65(3): 625−633
31. Burnham K P, Overton W S. Robust estimation of population size when capture probabilities vary among animals. Ecology, 1979, 60(5): 927−936
32. Heltshe J F, Forrester N E. Estimating species richness using the jackknife procedure. Biometrics, 1983, 39(1): 1−11
33. Smith E P, van Belle G. Nonparametric estimation of species richness. Biometrics, 1984, 40(1): 119−129
34. Efron B. Bootstrap methods: another look at the jackknife. In: Kotz S, Johnson N L, eds. Breakthroughs in Statistics: Methodology and Distribution. New York: Springer, 1992, 569–593
35. Horvitz D G, Thompson D J. A generalization of sampling without replacement from a finite universe. Journal of the American statistical Association, 1952, 47(260): 663−685
36. Särndal C E, Swensson B, Wretman J. Model Assisted Survey Sampling. New York: Springer, 2003
37. Chao A. Nonparametric estimation of the number of classes in a population. Scandinavian Journal of Statistics, 1984, 11(4): 265−270
38. Chao A, Shen T. User's guide for program spade (species prediction and diversity estimation). National Tsing Hua University, Dissertation, 2010
39. Chao A, Lee S M. Estimating the number of classes via sample coverage. Journal of the American statistical Association, 1992, 87(417): 210−217
40. Good I J. The population frequencies of species and the estimation of population parameters. Biometrika, 1953, 40(3-4): 237−264
41. Deolalikar V, Laffitte H. Extensive large-scale study of error in samping-based distinct value estimators for databases. 2016, arXiv preprint arXiv: 1612.00476
42. Motwani R, Vassilvitskii S. Distinct values estimators for power law distributions. In: Proceedings of the 3rd Workshop on Analytic Algorithmics and Combinatorics. 2006, 230–237
43. Korwar R M. On the observed number of classes from multivariate power series and hypergeometric distributions. Sankhyā: The Indian Journal of Statistics, Series B, 1988, 50(1): 39−59
44. Valiant P, Valiant G. Estimating the unseen: improved estimators for entropy and other properties. In: Proceedings of the 26th International Conference on Neural Information Processing Systems. 2013, 2157–2165
45. Li J, Lei R, Wang S, Wei Z, Ding B. Learning-based property estimation with polynomials. Proceedings of the ACM on Management of Data, 2024, 2(3): 1−27
46. Valiant G, Valiant P. Instance optimal learning of discrete distributions. In: Proceedings of the 48th Annual ACM Symposium on Theory of Computing. 2016, 142–155
47. Raghunathan A, Valiant G, Zou J. Estimating the unseen from multiple populations. In: Proceedings of the 34th International Conference on Machine Learning. 2017, 2855–2863
48. Valiant G, Valiant P. Estimating the unseen: improved estimators for entropy and other properties. Journal of the ACM (JACM), 2017, 64(6): 37
49. Valiant, G., & Valiant, P. (2010). Estimating the unseen: A sublinear-sample canonical estimator of distributions. Electronic Colloquium on Computational Complexity (ECCC) , TR10-180.
50. Acharya J, Das H, Orlitsky A, Suresh A T. A unified maximum likelihood approach for estimating symmetric properties of discrete distributions. In: Proceedings of the 34th International Conference on Machine Learning. 2017, 11–21
51. Pavlichin D S, Jiao J, Weissman T. Approximate profile maximum likelihood. Journal of Machine Learning Research, 2019, 20(122): 1−55
52. Acharya J, Das H, Mohimani H, Orlitsky A, Pan S. Exact calculation of pattern probabilities. In: Proceedings of 2010 IEEE International Symposium on Information Theory. 2010, 1498–1502
53. Orlitsky A, Sajama S, Santhanam N P, Viswanathan K, Zhang J. Algorithms for modeling distributions over large alphabets. In: Proceedings of International Symposium onInformation Theory. 2004, 304
54. Orlitsky A, Santhanam N, Viswanathan K, Zhang J. Theoretical and experimental results on modeling low probabilities. In: Proceedings of 2006 IEEE Information Theory Workshop – ITW '06 Punta del Este. 2006, 242–246
55. Vontobel P O. The bethe approximation of the pattern maximum

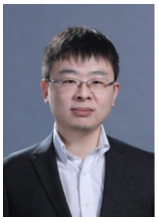
likelihood distribution. In: Proceedings of 2012 IEEE International Symposium on Information Theory Proceedings. 2012, 2012–2016

56. Vontobel P O. The bethe and sinkhorn approximations of the pattern maximum likelihood estimate and their connections to the valiant-valiant estimate. In: Proceedings of 2014 Information Theory and Applications Workshop (ITA). 2014, 1–10
57. Wu Y, Yang P. Minimax rates of entropy estimation on large alphabets via best polynomial approximation. IEEE Transactions on Information Theory, 2016, 62(6): 3702−3720
58. Wu Y, Yang P. Chebyshev polynomials, moment matching, and optimal estimation of the unseen. The Annals of Statistics, 2019, 47(2): 857−883
59. Chien I. Regularized weighted chebyshev approximations for support estimation. University of Illinois at Urbana-Champaign, Dissertation, 2019
60. Eden T, Indyk P, Narayanan S, Rubinfeld R, Silwal S, Wagner T. Learning-based support estimation in sublinear time. In: Proceedings of the 9th International Conference on Learning Representations. 2021
61. Wu R, Ding B, Chu X, Wei Z, Dai X, Guan T, Zhou J. Learning to be a statistician: learned estimator for number of distinct values. Proceedings of the VLDB Endowment, 2021, 15(2): 272−284
62. Whang K Y, Vander-Zanden B T, Taylor H M. A linear-time probabilistic counting algorithm for database applications. ACM Transactions on Database Systems (TODS), 1990, 15(2): 208−229
63. Swamidass S J, Baldi P. Mathematical correction for fingerprint similarity measures to improve chemical retrieval. Journal of Chemical Information and Modeling, 2007, 47(3): 952−964
64. Papapetrou O, Siberski W, Nejdl W. Cardinality estimation and dynamic length adaptation for bloom filters. Distributed and Parallel Databases, 2010, 28(2): 119−156
65. Alon N, Matias Y, Szegedy M. The space complexity of approximating the frequency moments. In: Proceedings of the 28th Annual ACM Symposium on Theory of Computing. 1996, 20–29
66. Cormode G. Count-min sketch. In: Liu L, Özsu M T, eds. Encyclopedia of Database Systems. New York: Springer, 2009, 511–516
67. Gibbons P B, Tirthapura S. Estimating simple functions on the union of data streams. In: Proceedings of the 13th Annual ACM Symposium on Parallel Algorithms and Architectures. 2001, 281–291
68. Bar-Yossef Z, Jayram T S, Kumar R, Sivakumar D, Trevisan L. Counting distinct elements in a data stream. In: Proceedings of the 6th International Workshop on Randomization and Approximation Techniques in Computer Science. 2002, 1–10
69. Beyer K, Gemulla R, Haas P J, Reinwald B, Sismanis Y. Distinct-value synopses for multiset operations. Communications of the ACM, 2009, 52(10): 87−95
70. Cohen E. Size-estimation framework with applications to transitive closure and reachability. Journal of Computer and System Sciences, 1997, 55(3): 441−453
71. Dasgupta A, Lang K J, Rhodes L, Thaler J. A framework for estimating stream expression cardinalities. In: Proceedings of the 19th International Conference on Database Theory. 2016
72. Flajolet P, Martin G N. Probabilistic counting algorithms for data base applications. Journal of Computer and System Sciences, 1985, 31(2): 182−209
73. Alon N, Matias Y, Szegedy M. The space complexity of approximating the frequency moments. Journal of Computer and System Sciences, 1999, 58(1): 137−147
74. Durand M, Flajolet P. Loglog counting of large cardinalities. In: Proceedings of the 11th Annual European Symposium on Algorithms. 2003, 605–617
75. Flajolet P, Fusy É, Gandouet O, Meunier F. HyperLogLog: the analysis of a near-optimal cardinality estimation algorithm. In: Proceedings of 2007 Conference on Analysis of Algorithms. 2007, 127–146
76. Hall A, Bachmann O, Büssow R, Gănceanu S, Nunkesser M. Processing a trillion cells per mouse click. Proceedings of the VLDB Endowment, 2012, 5(11): 1436−1446
77. Heule S, Nunkesser M, Hall A. HyperLogLog in practice: algorithmic engineering of a state of the art cardinality estimation algorithm. In: Proceedings of the 16th International Conference on Extending Database Technology. 2013, 683–692
78. Kane D M, Nelson J, Woodruff D P. An optimal algorithm for the distinct elements problem. In: Proceedings of the 29th ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems. 2010, 41–52
79. Ting D. Approximate distinct counts for billions of datasets. In: Proceedings of the 2019 International Conference on Management of Data. 2019, 69–86
80. Chiosa M, Preußer T B, Alonso G. SKT: a one-pass multi-sketch data analytics accelerator. Proceedings of the VLDB Endowment, 2021, 14(11): 2369−2382
81. Ertl O. SetSketch: filling the gap between MinHash and HyperLogLog. Proceedings of the VLDB Endowment, 2021, 14(11): 2244−2257
82. Cormode G, Yi K. Small Summaries for Big Data. Cambridge: Cambridge University Press, 2020
83. Li J, Wei Z, Ding B, Dai X, Lu L, Zhou J. Sampling-based estimation of the number of distinct values in distributed environment. In: Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining. 2022, 893–903
84. Charikar M, Chen K, Farach-Colton M. Finding frequent items in data streams. In: Proceedings of the 29th International Colloquium on International Colloquium on Automata, Languages, and Programming. 2002, 693–703
85. Wang P, Xie D, Zhao J, Li J, Li Z, Li R, Ren Y, Di J. Half-Xor: a fully-dynamic sketch for estimating the number of distinct values in big tables. IEEE Transactions on Knowledge and Data Engineering, 2024, 36(7): 3111−3125
86. Wang F, Chen Q, Li Y, Yang T, Tu Y, Yu L, Cui B. JoinSketch: a sketch algorithm for accurate and unbiased inner-product estimation. Proceedings of the ACM on Management of Data, 2023, 1(1): 81
87. Shekelyan M, Cormode G. Sequential random sampling revisited: hidden shuffle method. In: Proceedings of the 24th International Conference on Artificial Intelligence and Statistics. 2021, 3628–3636
88. Wei C, Salloum S, Emara T Z, Zhang X, Huang J Z, He Y. A two-stage data processing algorithm to generate random sample partitions for big data analysis. In: Proceedings of the 11th International Conference on Cloud Computing. 2018, 347–364
89. Salloum S, Huang J Z, He Y. Random sample partition: a distributed data model for big data analysis. IEEE Transactions on Industrial Informatics, 2019, 15(11): 5846−5854
90. Debnath S K, Dutta R. Secure and efficient private set intersection cardinality using bloom filter. In: Proceedings of the 18th International Conference on Information Security. 2015, 209–226
91. Guo D, Wu J, Chen H, Yuan Y, Luo X. The dynamic bloom filters. IEEE Transactions on Knowledge and Data Engineering, 2010, 22(1): 120−133
92. Ertl O. New cardinality estimation algorithms for HyperLogLog sketches. 2017, arXiv preprint arXiv: 1702.01284
93. Ertl O. UltraLogLog: a practical and more space-efficient alternative to HyperLogLog for approximate distinct counting. Proceedings of the VLDB Endowment, 2024, 17(7): 1655−1668
94. Tsan B, Datta A, Izenov Y, Rusu F. Approximate sketches. Proceedings of the ACM on Management of Data, 2024, 2(1): 66

95. Wang D, Pettie S. Better cardinality estimators for HyperLogLog, PCSA, and beyond. In: Proceedings of the 42nd ACM SIGMOD-SIGACT-SIGAI Symposium on Principles of Database Systems. 2023, 317–327
96. Pettie S, Wang D, Yin L. Non-mergeable sketching for cardinality estimation. In: Proceedings of the 48th International Colloquium on Automata, Languages, and Programming (ICALP 2021). 2021
97. Qiu Y, Wang Y, Yi K, Li F, Wu B, Zhan C. Weighted distinct sampling: cardinality estimation for SPJ queries. In: Proceedings of 2021 International Conference on Management of Data. 2021, 1465–1477
98. Dai B, Hu X, Yi K. Reservoir sampling over joins. Proceedings of the ACM on Management of Data, 2024, 2(3): 1−26
99. Kim K, Jung J, Seo I, Han W S, Choi K, Chong J. Learned cardinality estimation: an in-depth study. In: Proceedings of 2022 International Conference on Management of Data. 2022, 1214–1227
100. Kipf A, Kipf T, Radke B, Leis V, Boncz P A, Kemper A. Learned cardinalities: estimating correlated joins with deep learning. In: Proceedings of the 9th Biennial Conference on Innovative Data Systems Research. 2019
101. Sun J, Li G. An end-to-end learning-based cost estimator. Proceedings of the VLDB Endowment, 2019, 13(3): 307−319
102. Negi P, Wu Z, Kipf A, Tatbul N, Marcus R, Madden S, Kraska T, Alizadeh M. Robust query driven cardinality estimation under changing workloads. Proceedings of the VLDB Endowment, 2023, 16(6): 1520−1533
103. Hilprecht B, Schmidt A, Kulessa M, Molina A, Kersting K, Binnig C. DeepDB: learn from data, not from queries! Proceedings of the VLDB Endowment, 2020, 13(7): 992–1005
104. Yang Z, Kamsetty A, Luan S, Liang E, Duan Y, Chen X, Stoica I. NeuroCard: one cardinality estimator for all tables. Proceedings of the VLDB Endowment, 2020, 14(1): 61−73
105. Chien E, Milenkovic O, Nedich A. Support estimation with sampling artifacts and errors. In: Proceedings of 2021 IEEE International Symposium on Information Theory (ISIT). 2021, 244–249
106. Hao Y, Orlitsky A. Unified sample-optimal property estimation in near-linear time. In: Proceedings of the 33rd International Conference on Neural Information Processing Systems. 2019, 996
107. Shan J, Fu Y, Ni G, Luo J, Wu Z. Fast counting the cardinality of flows for big traffic over sliding windows. Frontiers of Computer Science, 2017, 11(1): 119−129

Jiajun LI is a PhD candidate at Gaolin School of Artificial Intelligence, Renmin University of China, China advised by Professor Zhewei Wei. He received his BE degree at School of Statistics, Renmin University of China, China in 2019. His research focuses on the approximate computing algorithm, AI for databases (AI4DB), and the design of data stream and sketch algorithms.

Runlin LEI is a PhD candidate at Gaolin School of Artificial Intelligence, Renmin University of China, China advised by Professor Zhewei Wei. He received his BE degree at School of Information and Technology, Shanghai University of Finance and Economics, China in 2022. His research focuses on graph neural networks and AI for databases.



Zhewei WEI is currently a professor at Gaoling School of Artificial Intelligence, Renmin University of China, China. He obtained his PhD degree at Department of Computer Science and Engineering, The Hong Kong University of Science and Technology (HKUST), China in 2012. He received the BSc degree in the School of Mathematical Sciences at Peking University, China in 2008. His research interests include graph algorithms, massive data algorithms, and streaming algorithms. He was the Proceeding Chair of SIGMOD/PODS2020 and ICDT2021, the Area Chair of ICML 2022/2023, NeurIPS 2022/2023, ICLR 2023, WWW 2023. He is also the PC member of various top conferences, such as VLDB, KDD, ICDE, ICML, and NeurIPS.