Efficient Approximation Framework for Attribute Recommendation (Technical Report)

Anonymous Author(s)

ABSTRACT

Trend analysis is a fundamental type of analytical query in online analytical processing (OLAP) systems. In trend analysis, a key step is to identify k valuable attributes whose distributions in two subsets under different predicates significantly differ for further investigation, where the difference is measured by metric functions. However, the exact solution that involves scanning all records is prohibitively expensive, particularly when handling large datasets in the era of big data. To minimize unnecessary data access, the existing state-of-the-art solution TopKAttr adopts sampling to avoid the expensive data scan. However, their solution still has two main drawbacks. Firstly, their solution is tailored only for two limited metric functions: the Earth Mover distance and Euclidean distance, and cannot be generalized to more complicated metric functions. Besides, their solution still aims to return the exact top-k answers via the sampling method, which still causes high running costs as shown in our experiment.

Motivated by these limitations, we propose a general approximation framework for attribute recommendation that efficiently returns the top-k attributes with theoretical guarantees while supporting an extensive range of metric functions, such as the Kolmogorov-Smirnov test (KS-test), Chebyshev distance, the Earth Mover distance, Euclidean distance, and with the potential to more metrics. The key to our framework is a new bound estimation strategy that can be applied to a wide spectrum of metrics, as we listed above. Based on our estimation framework, we further devise an efficient approximation algorithm with theoretical guarantees to answer the top-k queries, which is widely used in attribute recommendation. Extensive experiments on four real large datasets show that our framework gains up to an order of magnitude speed-up and consistently high accuracy compared to TopKAttr, providing a promising alternative for attribute recommendation in OLAP systems.

ACM Reference Format:

Anonymous Author(s). 2022. Efficient Approximation Framework for Attribute Recommendation (Technical Report). In *Proceedings of 2024 International Conference on Management of Data (SIGMOD '24).* ACM, New York, NY, USA, 18 pages. https://doi.org/XXXXXXX.XXXXXXX

1 INTRODUCTION

OLAP is widely used to analyze multidimensional data at a high speed on large volumes of data such as a data warehouse [6, 12, 18], data mart [6, 18], or some other unified centralized data stores [36, 44] from various perspectives for decision-making and problem-solving. Trend analysis [47] is an important type of analytical queries in OLAP systems, involving identifying patterns and trends in data over time. It finds extensive applications in business intelligence [24], financial analysis [17] and healthcare analytics [43]. In the era of big data, many companies need to deal with huge mining tables [11, 57] with hundreds or even thousands of attributes. To do trend analysis, and also most data analysis tasks, the first step is to identify a relatively small set of valuable attributes for further exploration and analysis since users cannot easily be aware of all the relevant attributes that can be used to answer their queries. These valuable attributes can be carefully selected by experienced data experts in each area by devising complicated rules or even choosing them manually. However, this leads to a heavy workload and requires specific skills and sufficient background knowledge in the related areas [54, 55]. To alleviate this burden, some automatic attribute recommendation systems have been proposed in the literature to analyze data and suggest valuable candidate attributes that are likely to be relevant to the query, helping users explore the data more comprehensively and gain deeper insights into the underlying patterns and relationships [47, 48, 55].

To find valuable attributes for trend analysis, the key observation of existing studies [47, 48, 55] is that the more significant the differences between two distributions are under different predicates, the more likely it is that such an attribute is important in trend analysis. For example, consider a database for an online shopping platform that stores customization records, including columns for gender, age group, login platform, region, timestamp, and more. Our goal is to analyse the sales trends between records in the morning and evening, i.e., records with different timestamps, to recommend personalized content. To achieve this, we search for attributes where the distribution among the records in the morning deviates significantly from that among records in the evening. Suppose we find that the variation on gender is larger than that on the other attributes. Hence gender is a valuable attribute in this trend analysis process and we can provide more personalized content based on gender during the corresponding time range. Two state-of-the-art solutions for attribute recommendation in trend analysis are SeeDB [47] and TopKAttr [48]. The main focus is to return a small set of valuable attributes for users according to their variations or distances between two distributions from the two subsets of records with respect to two ad hoc queries within a single huge table. The variations or distances are measured by metric functions such as the commonly used Kolmogorov–Smirnov test (KS-test) [16, 31], Chebyshev distance [1, 13], Earth Mover distance [47, 48], Euclidean distance [47, 48] and so on.

A straightforward approach to this problem is calculating the exact metric function score of each attribute by scanning all records. Existing OLAP systems are typically stored in a columnar format,

SIGMOD '24, June 2024, Santiago, Chile

ACM ISBN 978-1-4503-XXXX-X/18/06…$15.00
https://doi.org/XXXXXXX.XXXXXXX

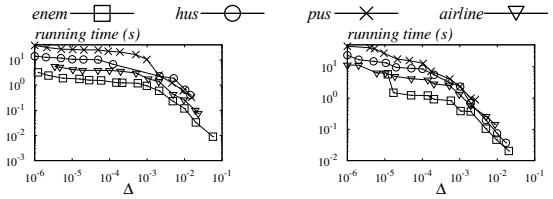
which makes data access more efficient. Nevertheless, the strategy of fully scanning records still leads to high latency for attribute recommendation, especially in large datasets with millions or even billions of records. SeeDB [47] employs sampling to reduce data access and query time with a heuristic multi-armed bandit pruning strategy. It creates an arm for each attribute, partitions the records, and updates the observed reward of each arm with the metric function score calculated from an unused partition step by step. Yet, the method mentioned above provides no theoretical guarantee and might return bad results.

TopKAttr [48] is the state-of-the-art solution for attribute recommendation that returns attributes using the sampling method with theoretical guarantees. The proposed solution only works for the Earth Mover distance and Euclidean distance. Assume that the distribution on attribute α of the records fulfilling predicate P_1 (resp. P_2) is $\mathbf{p}_1$ (resp. $\mathbf{p}_2$). To measure the deviation of $\mathbf{p}_1$ and $\mathbf{p}_2$ on attribute α, the Earth Mover distance (resp. Euclidean distance) takes the normalized L_1 (resp. L_2) norm of $\mathbf{p}_1 - \mathbf{p}_2$. However, their solution has two major disadvantages. Firstly, it supports limited metric functions, i.e., only Earth Mover distance and Euclidean distance but not other widely used metric functions such as the KS-test and Chebyshev distance. As we will explain in detail in Sec. 2.2, the main reason why TopKAttr cannot be generalized to more complicated metric functions is that it tries to bound the error holistically. In particular, take the Earth Mover distance as an example. Given the sampled set of records, assume that the distribution on attribute α of the sampled record fulfilling predicate P_1 (resp. P_2) is $\mathbf{q}_1$ (resp. $\mathbf{q}_2$). Then, the goal is to use $\|\mathbf{q}_1 - \mathbf{q}_2\|_1$ to approximate $\|\mathbf{p}_1 - \mathbf{p}_2\|_1$. In other words, TopKAttr aims to bound $|\|\mathbf{q}_1 - \mathbf{q}_2\|_1 - \|\mathbf{p}_1 - \mathbf{p}_2\|_1|$. By triangle inequality:

$$|\|\mathbf{q}_1 - \mathbf{q}_2\|_1 - \|\mathbf{p}_1 - \mathbf{p}_2\|_1| \leq \|\mathbf{p}_1 - \mathbf{q}_1\|_1 + \|\mathbf{p}_2 - \mathbf{q}_2\|_1$$

Intuitively, the more samples we have, the closer $\|\mathbf{p}_1 - \mathbf{q}_1\|_1$ and $\|\mathbf{p}_2 - \mathbf{q}_2\|_1$ are to zero, which provides a tight bound for the estimation to the exact Earth Mover distance. Hence, the goal is to derive a tight upper bound for $\|\mathbf{p}_1 - \mathbf{q}_1\|_1 + \|\mathbf{p}_2 - \mathbf{q}_2\|_1$. Next, TopKAttr further adopts concentration inequalities with sampling without replacement [14] to bound $\|\mathbf{p}_1 - \mathbf{q}_1\|_1$ and $\|\mathbf{p}_2 - \mathbf{q}_2\|_1$. We show how to bound $\|\mathbf{p}_1 - \mathbf{q}_1\|_1$, and the other term can be bounded similarly. Denote $\|\mathbf{p}_1 - \mathbf{q}_1\|_1$ as a function f_1^{EM} depending on the sampled records via sampling without replacement. Then, by existing concentration inequalities [14], TopKAttr shows that f_1^{EM} can be bounded close to $\mathbb{E}[f_1^{EM}]$ with an additive error of γ, i.e., $|f_1^{EM} - \mathbb{E}[f_1^{EM}]| \leq \gamma$. Then, they further show that $\mathbb{E}[f_1^{EM}]$ can be upper bounded explicitly depending on the number of sampled records if the metric function is Earth Mover distance. Denote the upper bound as $\overline{\mathbb{E}}[f_1^{EM}]$. Then, $\|\mathbf{p}_1 - \mathbf{q}_1\|_1 \leq \overline{\mathbb{E}}[f_1^{EM}] + \gamma$, Similarly, we can bound $\|\mathbf{p}_2 - \mathbf{q}_2\|_1$, and hence $|\|\mathbf{q}_1 - \mathbf{q}_2\|_1 - \|\mathbf{p}_1 - \mathbf{p}_2\|_1|$. To summarize, the above framework treats the metric function f_1^{EM} holistically, and a key step is to derive an explicit upper bound of $\mathbb{E}[f_1^{EM}]$ (and $\mathbb{E}[f_2^{EM}]$). Yet, deriving the upper bound of the expectation becomes challenging when the metric function is more complicated, limiting their applications.

Besides, TopKAttr still returns the exact top-k answers. They use the above sampling method to derive the lower- and upper-bound of the metric function scores. Notice that the more records are sampled, the tighter the bounds are. When the $(k+1)$-th smallest upper bound is larger than the k-th smallest lower bound, TopKAttr stops the sampling and returns the exact top-k answers. However, this still incurs high running costs and may cause a heavy workload when the difference Δ between the k-th and $(k+1)$-th largest scores is close. The query latency further increases when several attributes crowd near the k-th largest value, and it is extremely difficult to distinguish them. To better validate our motivation, we run the top-k query with $k = 16$ on four public datasets: enem, hus, pus, and airline, which are frequently used in existing attribute recommendation studies [47, 48] (More details of the datasets are in Section 6). We set $k = 16$ and choose 50 different queries. Notice that for different queries, the difference Δ of the k-th and $(k+1)$-th largest value differs. Thus, we can plot the running time of TopkAttr with the change of Δ among these 50 queries for the Earth Mover distance and Euclidean distance (that TopKAttr supports). Note that both the x-axis and y-axis are log-scale. Figure 1 shows that as the difference Δ of the k-th and $(k+1)$-th largest value decreases, the running cost of TopKAttr will increase by a large margin. This motivates us to present a more efficient top-k algorithm.



(a) Earth Mover Distance (b) Euclidean Distance

Figure 1: Impact of Δ: Running time of TopKAttr.

Main contributions. Motivated by the limitations of the states of the art, we present a general approximation framework for attribute recommendation, which efficiently returns the top-k attributes with theoretical guarantees. Our framework accommodates widely used metric functions, including the KS-test, Chebyshev distance, Earth Mover distance, and Euclidean distance, with the potential for more metrics as well. A key to the more generalized framework is that instead of treating the metric function holistically, we treat the sample probability of each attribute value as a random variable. For instance, for the Earth Mover distance, when considering $\|\mathbf{p}_1 - \mathbf{q}_1\|_1$, instead of considering it as a whole, we consider it separately for each attribute value. Suppose the support size, i.e., the number of distinct attribute values on attribute α is s_α. Then we consider these s_α dimensions in the distribution separately. As we now focus more on a fine-grained function, it becomes more feasible to derive estimation bounds. Nevertheless, deriving the bounds for the complicated metric functions is still challenging. We will show how to deal with complicated metric functions in Sec. 3. Then, we aggregate such estimation bounds together to derive the final error bound. We further present theoretical analysis to show how to apply the technique to other metric functions, showing the generalization ability of our proposed framework.

Moreover, using the derived lower- and upper-bounds of the estimated metric function scores, we devise an efficient algorithm that can answer approximate top-k queries for attribute recommendation with a success probability of $(1 - 1/n)$, where n is the total number of records. Initially, all attributes in the dataset are

included as candidates. During each iteration, we check whether the estimated top-k results with the current sample size meet the criteria of the approximate top-k query. The algorithm terminates if these conditions are satisfied. The approximate solution avoids spending too much sampling cost when the k-th and the $(k+1)$-th largest scores are close, improving the query efficiency while still providing strong theoretical guarantees on the returned results.

We validate the performance of our framework by conducting an experimental study on four real large datasets. The practical results demonstrate that our framework outperforms the alternatives on all datasets in all cases. Remarkably, our framework is up to three orders of magnitude faster than the exact solution and achieves up to one order of magnitude speed-up with consistently high accuracy compared to the state-of-the-art solution, providing a promising alternative for attribute recommendation in OLAP systems. Additionally, a user study on a real data analysis scenario demonstrates that there is no one-size-fits-all metric function, and a more general framework supporting multiple metric functions allows users to choose the function or combinations that best suit their needs, thereby avoiding the limitations of a specific metric function and potentially yielding better results.

2 PRELIMINARIES

2.1 Problem Statement

We consider a single table $\mathcal{T}$ with a set $A = \{\alpha_1, \alpha_2, \ldots, \alpha_d\}$ of d attributes. Let r be a record of $\mathcal{T}$ and let $r(\alpha)$ be the value of attribute α at record r. For an arbitrary attribute $\alpha \in A$, let s_α be the support size (or the number of distinct attribute values) of attribute α and assume that the distinct attribute values are $\{e_1, e_2, \ldots, e_{s_\alpha}\}$. Given two ad hoc queries Q_1 and Q_2, let $\mathcal{S}_1$ with a size of n_1 and $\mathcal{S}_2$ with a size of n_2 be the two subsets of $\mathcal{T}$ satisfying the predicates P_1 and P_2, respectively. Our target is to find attributes whose distributions in $\mathcal{S}_1$ and $\mathcal{S}_2$ have large deviations, where the deviation is measured by metric functions. Given an attribute α to be considered, let n_{1i} be the number of records in $\mathcal{S}_1$ whose attribute value on attribute α equals to e_i, i.e., $n_{1i} = |\{r | r \in \mathcal{S}_1 \wedge r(\alpha) = e_i\}|$. Then, n_{2i} can be defined similarly with respect to $\mathcal{S}_2$. Clearly, $n_1 = \sum_{i=1}^{s_\alpha} n_{1i}$ and $n_2 = \sum_{i=1}^{s_\alpha} n_{2i}$. To compare two distributions, define $p_{1i} = n_{1i}/n_1$ (resp. $p_{2i} = n_{2i}/n_2$) as the probability of a record r in $\mathcal{S}_1$ (resp. $\mathcal{S}_2$) so that $r(\alpha) = e_i$. We further define the distribution vectors $\mathbf{p}_1 = [p_{11}, p_{12}, \ldots, p_{1s_\alpha}]$ and $\mathbf{p}_2 = [p_{21}, p_{22}, \ldots, p_{2s_\alpha}]$. Given an attribute α, we focus on the following four metric functions to measure the deviation on the predicates P_1 and P_2.

Kolmogorov–Smirnov test (KS-test). As claimed by Engmann et al. [16], the KS-test is one of the most commonly used tests of distributions. Specifically, KS-test derives the empirical cumulative distribution functions of two distributions on a given attribute α and then calculates their distance [31], which is defined as

$$D_{KS}(\alpha) = \max_{i \in [1, s_\alpha]} |F_{1i} - F_{2i}| = \|\mathbf{F}_1 - \mathbf{F}_2\|_\infty,$$

where $F_{1i} = \sum_{k=1}^{i} p_{1k}$ and $F_{2i} = \sum_{k=1}^{i} p_{2k}$ for $i \in [1, s_\alpha]$ are the empirical cumulative distribution functions of the first and the second distribution on attribute α respectively. In addition, we have $\mathbf{F_1} = [F_{11}, F_{12}, \ldots, F_{1s_\alpha}]$ and $\mathbf{F_2} = [F_{21}, F_{22}, \ldots, F_{2s_\alpha}]$. For example, assume that we consider two distributions $\mathbf{p}_1 = [0.1, 0.2, 0.3, 0.4]$ and $\mathbf{p}_2 = [0.4, 0.3, 0.1, 0.2]$ for attribute α where $s_\alpha = 4$. Their empirical cumulative distributions can be derived as $\mathbf{F}_1 = [0.1, 0.3, 0.6, 1.0]$ and $\mathbf{F}_2 = [0.4, 0.7, 0.8, 1.0]$. KS-test uses the largest absolute difference among all position $i \in [1, 4]$ between the empirical cumulative distributions $\mathbf{F}_1$ and $\mathbf{F}_2$ to indicate the distance between $\mathbf{p}_1$ and $\mathbf{p}_2$. Then $D_{KS}(\alpha) = |F_{12} - F_{22}| = 0.4$ as the absolute difference between the second elements of $\mathbf{F}_1$ and $\mathbf{F}_2$ is the largest.

Chebyshev distance. Chebyshev distance is another metric of distribution comparison [1, 13], which is defined by the largest absolute probability difference among any coordinate dimensions. Specifically, for attribute α,

$$D_{CH}(\alpha) = \max_{i \in [1, s_\alpha]} |p_{1i} - p_{2i}| = \|\mathbf{p}_1 - \mathbf{p}_2\|_\infty.$$

Following the above example with the same distributions $\mathbf{p}_1$ and $\mathbf{p}_2$, $D_{CH}(\alpha) = |p_{11} - p_{21}| = 0.3$.

Earth Mover distance & Euclidean distance. We also include Earth Mover distance and Euclidean distance in our consideration as they are effective metrics of deviation [47, 48]. We have earth Mover distance between two distributions on attribute α as $D_{EM}(\alpha) = \frac{1}{2}\sum_{i=1}^{s_\alpha} |p_{1i} - p_{2i}| = \frac{1}{2}\|\mathbf{p}_1 - \mathbf{p}_2\|_1$ and the corresponding Euclidean distance $D_{EU}(\alpha) = \sqrt{\frac{1}{2}\sum_{i=1}^{s_\alpha}(p_{1i} - p_{2i})^2} = \frac{1}{\sqrt{2}}\|\mathbf{p}_1 - \mathbf{p}_2\|_2$. Both multipliers $\frac{1}{2}$ and $\frac{1}{\sqrt{2}}$ are used to ensure that all distance metric values fall in the range $[0, 1]$.

Attribute recommendation. Using the metric functions above (but not limited to), we can now formalize attribute recommendation problems. When the user specifies two predicates for two ad hoc queries with a metric function, the attribute recommendation system analyses the underlying records and returns several valuable attributes of interest. Following previous works [47, 48], we define the top-k query as follows.

Definition 1 (Top-k Query for Attribute Recommendation). *Given a table $\mathcal{T}$ that includes a set A of attributes, let D be the provided metric function and let $\mathcal{S}_1$ and $\mathcal{S}_2$ be the subsets of $\mathcal{T}$ that match predicate P_1 and P_2 respectively. The top-k query returns a set R of k attributes such that $D(\alpha) \geq D(\alpha')$ for any $\alpha \in R$ and $\alpha' \in A \setminus R$.*

The top-k query serves an important role as a pre-filtering process to prune most irrelevant attributes, which significantly alleviates the time for attribute selection. Practically, the approximate top-k query is sufficient, as seen in applications such as subtree matching [3], binary pattern discovery [30], keyword proximity search [26], anomaly detection [28], mobile information subscription [7], object-class retrieval [37], personalized PageRank [22, 23, 50–53], data statistics like empirical entropy [8] and empirical variance [9]. Therefore, it would be valuable to design a top-k algorithm with an approximation parameter that can quickly return an approximate result when a strict requirement is not necessary. The algorithm can still return the exact result by adjusting the parameter setting when needed. Thus, we define the approximate top-k query as follows.

Definition 2 (Approximate Top-k Query). *Given a metric function D, two subsets $\mathcal{S}_1$ and $\mathcal{S}_2$ of records with attributes in a set A, a positive integer k and a relative error ϵ, the approximate top-k query returns a set R of k attributes such that the following two conditions*

- $\left|D(\alpha_i') - \hat{D}(\alpha_i')\right| \leq \epsilon$ *for any* $i \in [1, k]$
- $\left|D(\alpha_i^*) - \hat{D}(\alpha_i')\right| \leq \epsilon$ *for any* $i \in [1, k]$

Table 1: Frequently used notations.

Notation	Description
$\mathcal{T}$, n	The table $\mathcal{T}$ and the number of records in $\mathcal{T}$
$\mathcal{S}_1$ / $\mathcal{S}_2$	The subset of $\mathcal{T}$ fulfilling predicate P_1 / P_2
n_1 / n_2	The number of records in $\mathcal{S}_1$ / $\mathcal{S}_2$
d	The number of attributes in $\mathcal{T}$
m_1 / m_2	The number of samples from $\mathcal{S}_1$ / $\mathcal{S}_2$
A	The set of all attributes in $\mathcal{T}$
α,s_α	An attribute from set A and its support size
p_f	The failure probability of the algorithm
$D(\alpha)$	The metric function value of α
$\hat{D}(\alpha)$	The estimation of $D(\alpha)$
$\underline{D}(\alpha), \overline{D}(\alpha)$	The lower- and upper-bound of $D(\alpha)$
ϵ	The error bound for the approximate top-k query
η	The smaller selectivity for predicates P_1 and P_2

are satisfied with at least $1 - p_f$ *probability where* $\alpha'_1, \alpha'_2, \ldots, \alpha'_k$ *are the returned attributes and* α^*_i *is the attribute with the exact* i*-th largest metric function value.*

In the above definition, the first condition requires that the estimated metric function values of the returned attributes are close to their exact values. The second condition makes sure that the returned k attributes have similar values to the exact k largest values. Combining these two conditions ensures that the returned approximate top-k results will have high quality. The degree of closeness is measured by a predefined parameter ϵ. Intuitively, the approximate top-k query provides a trade-off between accuracy and efficiency through the user setting of absolute error ϵ. Additionally, the frequently used notations are listed in Tab. 1.

2.2 Existing Solutions

Exact solution. A straightforward solution is to scan all records satisfying the predicates in subsets $\mathcal{S}_1$ and $\mathcal{S}_2$ column by column (supposing that data are stored in a columnar format) and calculate the exact metric function values for all attributes. Although modern databases such as SQL Server [2] support columnar storage for high efficiency of OLAP queries, it still leads to high costs while there are millions of records in subsets for ad hoc queries, which is common in modern data warehouses, such as Snowflake [12], Amazon Redshift [18] and IBM Db2 Warehouse [6].

Existing approximate solutions. Approximate solutions use sampled records to approach exact metric function values. Sampling-based approximations avoid scanning all records, making them more efficient than the exact solution when the data consists of millions or even billions of records with tens to hundreds of attributes and data access is the bottleneck [47, 48].

Nonetheless, there are some obstacles when approximating metric functions with sampling. Standard concentration inequalities such as Hoeffding's inquality [21] and McDiarmid's inequality [32] cannot be directly adopted to estimate the metric function since they require that the function value can be expressed as the average of samples. SeeDB [47] is a data-driven visualization framework that uses worst-case confidence intervals and a heuristic multi-armed bandit strategy to prune attributes with low metric function values. It creates an arm for each attribute, partitions the records, and updates the observed reward of each arm with the metric function value calculated step by step from an unused partition. Each estimate obtained from the iteration is regarded as a sample of the approximate metric function value. However, since the expectation of these sampled values from each partition does not match the metric function calculated from the entire dataset, the derived confidence intervals are biased. This bias can cause the returned k attributes to deviate significantly from the exact top-k attributes, without any probabilistic guarantee.

To provide results with probabilistic guarantees, Wang et al. propose TopKAttr [48], which is the state-of-the-art solution for attribute recommendation. In practical applications, data partitioning is a commonly used technique in column-oriented database systems that provide single-machine in-memory deployment and data partitioning, such as Microsoft SQL Server [2], MonetDB [5], and SAP HANA [42]. It is also used in distributed data platforms such as data warehouses [45] and Hadoop [41]. TopKAttr focuses on the single machine in-memory setting and considers the situation where records in table $\mathcal{T}$ are randomly partitioned. In this case, each record can be modeled as a sample unit without replacement from table $\mathcal{T}$. For analyzing the sampling without replacement model, define the permutation of records in subset $\mathcal{S}_1$ (resp. $\mathcal{S}_2$) as $\Pi_1 = (\pi_1, \pi_2, \ldots, \pi_{n_1})$ (resp. $\Pi_2 = (\pi_1, \pi_2, \ldots, \pi_{n_2})$) where n_1 (resp. n_2) is the number of records in $\mathcal{S}_1$ (resp. $\mathcal{S}_2$) underlying predicate P_1 (resp. P_2). Each value π_i in Π_1 (resp. Π_2) defines the index of the i-th sample among n_1 (resp. n_2) records where $i \in [1, n_1]$ (resp. $[1, n_2]$). Consider a fixed attribute α. Without loss of generality, assume that there are m_1 (resp. m_2) records sampled from $\mathcal{S}_1$ (resp. $\mathcal{S}_2$). Then define m_{1i} (resp. m_{2i}) as the number of records in $\mathcal{S}_1$ (resp. $\mathcal{S}_2$) whose attribute value on attribute α equals to e_i among these m_1 (resp. m_2) samples, and we have $m_1 = \sum_{i=1}^{s_\alpha} m_{1i}$ (resp. $m_2 = \sum_{i=1}^{s_\alpha} m_{2i}$). Then we define $q_{1i} = m_{1i}/m_1$ (resp. $q_{2i} = m_{2i}/m_2$) as the probability that the record has an attribute value of e_i on attribute α among the m_1 (resp. m_2) samples for $i \in [1, s_\alpha]$. Similar to the definitions of $\mathbf{p}_1$ and $\mathbf{p}_2$, we define vectors $\mathbf{q}_1 = [q_{11}, q_{12}, \ldots, q_{1s_\alpha}]$ and $\mathbf{q}_2 = [q_{21}, q_{22}, \ldots, q_{2s_\alpha}]$ for the sampling results. With these sampled records from $\mathcal{S}_1$ and $\mathcal{S}_2$, TopKAttr calculates the estimation of Earth Mover distance (resp. Euclidean distance) by $\hat{D}_{EM}(\alpha) = \frac{1}{2}\|\mathbf{q}_1 - \mathbf{q}_2\|_1$ (resp. $\hat{D}_{EU}(\alpha) = \frac{1}{\sqrt{2}}\|\mathbf{q}_1 - \mathbf{q}_2\|_2$) for each attribute. However, the approximate results, i.e., $\hat{D}_{EM}(\alpha)$ and $\hat{D}_{EU}(\alpha)$, still have no theoretical guarantee for their accuracy.

We then rephrase how TopKAttr provides results for Earth Mover distance and Euclidean distance with probabilistic guarantees. Since the confidence radius derivations of these two distances are similar, we only take the analysis of Earth Mover distance as an illustration here. Consider the absolute difference between the exact Earth Mover distance value $D_{EM}(\alpha)$ and its estimation $\hat{D}_{EM}(\alpha)$, i.e., $|D_{EM}(\alpha) - \hat{D}_{EM}(\alpha)|$. The less the absolute difference is, the more accurate the estimation result is. Then TopKAttr derives an upper-bound of this difference as the sum of two parts, i.e., $|D_{EM}(\alpha) - \hat{D}_{EM}(\alpha)| \le \frac{1}{2}f_1^{EM} + \frac{1}{2}f_2^{EM}$, where $f_1^{EM} = \|\mathbf{p}_1 - \mathbf{q}_1\|_1$ and $f_2^{EM} = \|\mathbf{p}_2 - \mathbf{q}_2\|_1$. Note that each part $\frac{1}{2}f_j^{EM}$ for $j = 1, 2$ only concerns the data distribution on attribute α in subset $\mathcal{S}_j$ and the corresponding distribution of the samples from the same set. If f_1^{EM} and f_2^{EM} can be bounded probabilistically, $|D_{EM}(\alpha) - \hat{D}_{EM}(\alpha)|$ can

also be bounded. To obtain an upper-bound of f_j^{EM} for $j = 1, 2$ with concentration inequalities, we require the expectation $\mathbb{E}[f_j^{EM}]$ of f_j^{EM} or its upper bound $\overline{\mathbb{E}}[f_j^{EM}]$. TopKAttr shows that the upper bounds $\overline{\mathbb{E}}[f_j^{EM}]$ and $\overline{\mathbb{E}}[f_j^{EU}]$ can be derived for L_1 norm and L_2 norm. However, it is unclear how to derive such an expectation for more complex metric functions, such as KS-test and Chebyshev distance. The limitations of the techniques used in TopKAttr motivate us to devise a more general approach.

Looking back to TopKAttr, it employs a concentration inequality for a transductive learning problem in [14] to bound f_1^{EM} (resp. f_2^{EM}) with the upper-bound $\overline{\mathbb{E}}[f_1^{EM}]$ (resp. $\overline{\mathbb{E}}[f_2^{EM}]$) of $\mathbb{E}[f_1^{EM}]$ (resp. $\mathbb{E}[f_2^{EM}]$). Then TopKAttr derives the confidence radius of $D_{EM}(\alpha)$ with union bound. When the context is clear, we omit the bracket including attribute α for space saving. The lower- and upper-bounds are shown below.

Lemma 1 (Theorem 4 in [48]). *Given attribute α whose support size is s_α, m_1 (resp. m_2) sampled records from subset $\mathcal{S}_1$ (resp. $\mathcal{S}_2$), and a failure probability p_f, we have that both*

$$\underline{D}_{EM} = \hat{D}_{EM} - r_{EM} \text{ and } \overline{D}_{EM} = \hat{D}_{EM} + r_{EM}$$

hold with probability at least $1 - p_f$, where

$$r_{EM} = \sum_{j=1}^{2} \left(\overline{\mathbb{E}}[f_j^{EM}] + h_j \sqrt{2 \log \left(\frac{\sqrt{h_1} + \sqrt{h_2}}{p_f \sqrt{h_j}} \right)} \right)$$

and $h_j = \sqrt{(n_j - m_j)/m_j/n_j}$.

Then, TopKAttr utilizes the bounds of each D_{EM} obtained from the initial samples and identifies competing attributes whose confidence intervals overlap. For these competing attributes, more samples are considered to calculate a tighter bound. The competing attributes are updated based on the new bounds in each iteration. The algorithm only terminates when the exact top-k attributes are completely separated from the other attributes, meaning their confidence intervals do not overlap. Although TopKAttr considers the approximate result of each attribute, its mechanism still requires the exact top-k attributes, which can be expensive when the k-th and $(k+1)$-th largest metric function values for the corresponding attributes are close. Besides, there may be more than two borderline attributes, which increases the computation cost for distinguishing them precisely. In contrast, obtaining approximate top-k results with theoretical guarantees is sufficient since it does not matter which specific borderline attribute is returned.

3 APPROXIMATION OF METRIC FUNCTIONS

From a high-level perspective, our proposed framework aims to return the approximate top-k attributes with theoretical guarantees efficiently, supporting commonly used tests such as KS-test, Chebyshev distance, Earth Mover distance, Euclidean distance, and potentially more metrics. The most challenging part of our framework is devising a general technique to approximate the above-mentioned metric functions. The state-of-the-art solution for attribute recommendation, i.e., TopKAttr, only supports Earth Mover distance and Euclidean distance. They directly regard each metric function as a whole and derive the upper bound of the estimated error with the inequality in [14]. However, the techniques in TopKAttr cannot be extended to other metric functions, such as KS-test and Chebyshev distance. To fill this gap, we carefully devise a nontrivial general approach to estimating metric functions. Instead of directly estimating the metric function as a whole, we zoom in on each dimension of the distribution, which is more flexible. Specifically, we derive the estimated bound of each dimension with high probability and then aggregate these estimations with a union bound. The specific aggregation function depends on the required metric function, e.g., summation for Earth Mover distance and taking the maximum for Chebyshev distance from their expressions. As our framework is devised based on each dimension that almost all metric functions for comparing distributions essentially focus on, it is more applicable to general metric functions. In practice, our technique can approximate many metric functions such as KS-test, Chebyshev distance, Earth Mover distance, Euclidean distance, and potentially more metrics, overcoming the limitations of TopKAttr.

Following the setting in [48] and [49], the records in table $\mathcal{T}$ are randomly partitioned and each record can be modeled as a sampling unit without replacement, which is common in column-oriented database systems that provide single-machine in-memory deployment and data partitioning such as Microsoft SQL Server [2], MonetDB [5], and SAP HANA [42] as well as distributed data platforms such as data warehouse [45] and Hadoop [41]. Our framework focuses on in-memory column store layouts and the overview is as follows. Using initial samples, calculate the confidence interval of each candidate attribute corresponding to the metric function for the query. Then prune attributes that will not be returned with high probability. For the remaining attributes, we adaptively increase the sample size and derive narrower confidence intervals, where the pruning procedure is conducted synchronously. The algorithm terminates until the found attributes satisfy the conditions of the approximate top-k query, which are proposed to accelerate the query with theoretical guarantees. In this section, we will focus on the proposed technique to approximate metric functions since it is the core component of our framework. The details of approximate top-k query processing will be deferred to the next section.

3.1 Lower- and Upper-Bounds of KS-Test

To demonstrate the proposed technique for computing confidence intervals of metric functions, we will start with KS-test. Later, we will show how to extend to other metric functions.

Some definitions for estimating KS-test. Before conducting the analysis, we provide some definitions for estimating the KS-test. Recall that F_{ij} for $i \in [1, s_\alpha]$ is the empirical cumulative distribution on attribute α of records in subset $\mathcal{S}_j$ where $j = 1, 2$. Besides, q_{1i} (resp. q_{2i}) is the probability that the value on attribute α of the record within the m_1 (resp. m_2) samples equals to e_i where $i \in [1, s_\alpha]$. Then define $G_{1i} = \sum_{k=1}^{i} q_{1k}$ (resp. $G_{2i} = \sum_{k=1}^{i} q_{2k}$) for $i \in [1, s_\alpha]$ as the empirical cumulative distribution function based on m_1 (resp. m_2) samples on attribute α where s_α is the support size of attribute α. We also define $\mathbf{G}_1 = [G_{11}, G_{12}, \ldots, G_{1s_\alpha}]$ and $\mathbf{G}_2 = [G_{21}, G_{22}, \ldots, G_{2s_\alpha}]$ for ease of discussion later. Then, the estimation of the KS-test for attribute α based on the sampling results is $\hat{D}_{KS}(\alpha) = \max_{i \in [1, s_\alpha]} |G_{1i} - G_{2i}| = \|\mathbf{G}_1 - \mathbf{G}_2\|_\infty$.

Next, we will bound the absolute difference between the exact KS-test $D_{KS}(\alpha)$ and its estimation $\hat{D}_{KS}(\alpha)$ on attribute α, i.e., $|D_{KS}(\alpha) - \hat{D}_{KS}(\alpha)|$, with probabilistic guarantees. Our main idea is to analyze each dimension of the empirical cumulative distribution function and limit the discrepancy between the actual and its estimated value with a high probability. As KS-test asks for the largest deviation among all dimensions, we can bound the gap between the true KS-test and its estimation through the maximum gap in each dimension using a union bound, where the detailed analysis is as follows. With the triangle inequality for the infinity norm, we have that $|D_{KS}(\alpha) - \hat{D}_{KS}(\alpha)| = |\|\mathbf{F}_1 - \mathbf{F}_2\|_\infty - \|\mathbf{G}_1 - \mathbf{G}_2\|_\infty|$ satisfies:

$$|D_{KS}(\alpha) - \hat{D}_{KS}(\alpha)| \leq \|\mathbf{F}_1 - \mathbf{G}_1\|_\infty + \|\mathbf{F}_2 - \mathbf{G}_2\|_\infty. \quad (1)$$

Define $f_j^{KS} = \|\mathbf{F}_j - \mathbf{G}_j\|_\infty$ for $j = 1, 2$, where f_j^{KS} is the L_∞ distance between the actual empirical cumulative distribution $\mathbf{F}_j$ on attribute α in subset $\mathcal{S}_j$ and the corresponding cumulative distribution $\mathbf{G}_j$ for the m_j sampled records from $\mathcal{S}_j$. When the context is clear, we omit the superscript of f_j^{KS}. What we need to do next is to bound f_j for $j = 1, 2$, based on sampling. If we can bound f_j, we can bound the error of $|D_{KS}(\alpha) - \hat{D}_{KS}(\alpha)|$ by Eqn. 1, achieving our goal.

Establishing lower- and upper-bounds of KS-test. We then show how to bound f_1 (resp. f_2) via sampling without replacement. Note that sampling without replacement can be regarded as first deriving a random permutation Π and then sampling sequentially from this permutation. As function f_j depends on the permutation of sampled records satisfying the predicates, we define $f_j(\Pi_j)$ as the value of f_j with respect to the first m_j samples from a random permutation Π_j. Also, we define Π_j^{rs} as the permutation when swapping π_r and π_s in Π_j, where $r \in [1, m_j]$ and $s \in [m_j + 1, n_j]$. We then adopt a concentration inequality for sampling without replacement [15] to derive the bound, or called confidence radius, of $D_{KS}(\alpha)$, where the inequality is rephrased as follows.

Lemma 2 ([15]). *Let Π_j be a random permutation and $f_j(\Pi_j)$ be a function with $|f(\Pi_j) - f(\Pi_j^{r,s})| < \beta$ for all $r \in [1, m_j]$ and $s \in [m_j + 1, n_j]$. Then, we have that*

$$\mathbb{P}\left(f_j\left(\Pi_j\right) - \mathbb{E}[f_j\left(\Pi_j\right)] \geq \gamma\right) \leq \exp\left(-\frac{2\gamma^2}{m_j\beta^2}\left(\frac{n_j - 1/2}{n_j - m_j}\right)\left(1 - \frac{1}{2\max\left(m_j, n_j - m_j\right)}\right)\right).$$

To apply the above lemma, we first need to derive an upper bound of $|f_j(\Pi_j) - f_j(\Pi_j^{rs})|$, i.e., the absolute difference of f_j when swapping a pair of values on attribute α within the sampled and unsampled records in subset $\mathcal{S}_j$ for $j = 1, 2$. When defining $l = \arg\max_{i \in [1, s_\alpha]} |F_{ji} - G_{ji}|$, we have

$$\left|f_j(\Pi_j) - f_j(\Pi_j^{rs})\right| \leq \left|\left|\frac{\sum_{k=1}^{l} n_{jk}}{n_j} - \frac{\sum_{k=1}^{l} m_{jk}}{m_j}\right| - \left|\frac{\sum_{k=1}^{l} n_{jk}}{n_j} - \frac{\sum_{k=1}^{l} m_{jk} - 1}{m_j}\right|\right| \leq \frac{1}{m_j}.$$

To derive a bound of f_j, it is also necessary to know $\mathbb{E}[f_j\left(\Pi_j\right)]$ of f_j or, alternatively, an upper bound of $\mathbb{E}[f_j\left(\Pi_j\right)]$, which is nontrivial to derive for KS-test and Chebyshev distance. Nevertheless, here we assume that we already have an upper bound $\overline{\mathbb{E}}[f_j^{KS}]$ of $\mathbb{E}[f_j^{KS}]$, where this assumption will be resolved in the next section shortly. Given the upper bound of $\mathbb{E}[f_j^{KS}]$, we can then derive the upper bound of f_j. Given the upper bound of f_j, we can then bound $|D_{KS}(\alpha) - \hat{D}_{KS}(\alpha)|$ via Eqn. 1. By applying Lem. 2 and Eqn. 1, we can derive the confidence radius of $D_{KS}(\alpha)$ as shown below.

Lemma 3. *Given attribute α whose support size is s_α, m_1 (resp. m_2) sampled records from subset $\mathcal{S}_1$ (resp. $\mathcal{S}_2$), and a failure probability p'_f, we have that both*

$$\underline{D}_{KS} = \hat{D}_{KS} - r_{KS} \text{ and } \overline{D}_{KS} = \hat{D}_{KS} + r_{KS}$$

hold with probability at least $1 - p'_f$, where

$$r_{KS} = \sum_{j=1}^{2}\left(\overline{\mathbb{E}}[f_j^{KS}] + h_j\sqrt{2\ln\left(\frac{h_1 + h_2}{p'_f \cdot h_j}\right)}\right)$$

and $h_j = \sqrt{\frac{n_j - m_j}{4(n_j - \frac{1}{2})m_j(1 - \frac{1}{2\max(m_j, n_j - m_j)})}}$.

Lem. 3 provides lower- and upper-bounds of KS-test based on samples with theoretical guarantees. These bounds can be applied to the approximate top-k query in our framework. Besides, we will show the optimization technique to tighten the bounds in Sec. 3.3.

3.2 Upper Bounding the Expectation of f_j^{KS}

Remind that we assume a bound $\overline{\mathbb{E}}[f_j^{KS}]$ for the expectation $\mathbb{E}[f_j^{KS}]$ where $f_j^{KS} = \|\mathbf{F}_j - \mathbf{G}_j\|_\infty$ for $j = 1, 2$. The derivation of this bound is challenging and will be established in this section.

Establishing bounds on the expectation of f_j^{KS}. Recap that in TopKAttr, to bound f_j, it derives an error bound γ between f_j and $\mathbb{E}[f_j]$, i.e., $|f_j - \mathbb{E}[f_j]| \leq \gamma$. Then, it further derives an upper bound $\overline{\mathbb{E}}[f_j]$ of $\mathbb{E}[f_j]$ to obtain the bound of f_j. However, the upper-bound $\overline{\mathbb{E}}[f_j]$ of $\mathbb{E}[f_j]$ cannot be computed explicitly with the technique in TopKAttr. Consequently, we solve this problem from another direction, which is the key part of our technique. According to the definition of expectation, $\mathbb{E}[f_j] = \sum_{k=1}^{w} x_k u_k$ when regarding f_j as a random variable where x_k is a possible outcome of f_j, u_k is the corresponding probability, and there are in total w outcomes. Without loss of generality, we assume that the index of each possible outcome corresponds to the ascending order of all outcomes, i.e., $x_k < x_{k+1}$ for $k \geq 1$. Since $f_j \in [0, 1]$ from the definitions of metric functions, x_k belongs to the same range, i.e., $x_k \in [0, 1]$. However, this sum is difficult to calculate since we need to consider all possible sampling results. Alternatively, we consider separating consecutive outcomes x_k into finite groups. Define I_g as the set of indices of outcomes and their corresponding probabilities falling into group g. For group g, we represent it with an upper-bound y_g of all outcomes x_k in this group, i.e., $y_g = \max_{l \in I_g} x_l$. Also, we sum up all the corresponding probabilities u_k in group g as v_g, i.e., $v_g = \sum_{l \in I_g} u_l$. To control the number t of groups and ensure that the sum of all probability are 1, we define t as $\lceil \log_2 n_j \rceil + 1$ and the corresponding probabilities for t groups are set as $v_g = \frac{1}{2^g}$ for each group $g \in [1, t-1]$ and $v_t = \frac{1}{2^{t-1}}$ such that $\sum_{g=1}^{t} v_g = \frac{1}{2} + \frac{1}{4} + \cdots + \frac{1}{2^{t-2}} + \frac{2}{2^{t-1}} = 1$, where the last item $\frac{2}{2^{t-1}}$ explains two equal probabilities v_{t-1} and v_t. Therefore, we have $\mathbb{E}[f_j] = \sum_{k=1}^{w} x_k u_k = \sum_{g=1}^{t} \sum_{l \in I_g} x_l u_l \leq \sum_{g=1}^{t} \max_{l' \in I_g} x_{l'} \sum_{l \in I_g} u_l \leq \sum_{g=1}^{t} y_g v_g$.

Algorithm 1: Upper-Bound of Expectation

Input: An attribute α, m_j sampled records from subset $\mathcal{S}_j$ of $\mathcal{T}$ for $j = 1, 2$
Output: An upper-bound $\overline{\mathbb{E}}[f_j]$ of the expectation $\mathbb{E}[f_j]$

1 Set a series of probabilities of success $v'_1 \leftarrow \frac{1}{2}, v'_2 \leftarrow \frac{3}{4}, \ldots, v'_{t-1} \leftarrow 1 - \frac{1}{2^{t-1}}, v'_t \leftarrow 1$ where $v'_g \leftarrow 1 - \frac{1}{2^g}$ for $g \in [1, t-1]$ and $t = \lceil \log_2 n_j \rceil + 1$;
2 **for** $g \in [1, t-1]$ **do**
3 **for** $i \in [1, s_\alpha]$ **do**
4 Calculate the upper-bound y_{gi} of $|F_{ji} - G_{ji}|$ by Lem. 4 with v'_g;
5 $y_g \leftarrow \max_{i \in [1, s_\alpha]} y_{gi}$
6 $y_t \leftarrow 1$;
7 $v_g \leftarrow v'_g$ for $g = 1$ and $v_g \leftarrow v'_g - v'_{g-1}$ for $g \in [2, t]$;
8 $\overline{\mathbb{E}}[f_j] \leftarrow \sum_{g=1}^{t} y_g v_g$;
9 **return** $\overline{\mathbb{E}}[f_j]$;

Example 1. *Assume that we have that the outcome $x_1 = 0.1$ has probability $u_1 = 0.1$, $x_2 = 0.3$ with $u_2 = 0.4$, $x_3 = 0.4$ with $u_3 = 0.4$, and $x_4 = 0.5$ with $u_4 = 0.1$. Then, a possible group is as follows: When the total number of groups is 3, we can set a series of probabilities as $v_1 = \frac{1}{2}$, $v_2 = \frac{1}{4}$ and $v_3 = \frac{1}{4}$. We then group outcomes x_1 and x_2 as group 1 since $u_1 + u_2 = v_1 = 0.5$, i.e., $I_1 = \{1, 2\}$, where $y_1 = \max(x_1, x_2) = 0.3$. As the probability u_3 is larger than the probability v_2 of the second group, i.e., $u_3 > \frac{1}{4}$, x_3 should contribute to groups 2 and 3. We then have group 2, $I_2 = \{3\}$ where $y_2 = x_3 = 0.4$ and group 3, $I_3 = \{3, 4\}$ where $y_3 = \max(x_3, x_4) = 0.5$. Then, $\mathbb{E}[f_j] \le \sum_{g=1}^{3} y_g v_g = 0.3 \times \frac{1}{2} + 0.4 \times \frac{1}{4} + 0.5 \times \frac{1}{4} = 0.375$.*

A new question arises: how to set up groups properly and derive each y_g corresponding to the group? For this, we zoom in on the expression of f_j. We aim to derive an upper-bound of f_j, i.e., $\|\mathbf{F}_j - \mathbf{G}_j\|_\infty$, for a given probability v'_g where v'_g is defined as $v'_g = \sum_{l=1}^{g} v_l$. Specifically, $v'_1 = \frac{1}{2}, v'_2 = \frac{3}{4}, \ldots, v'_{t-1} = 1 - \frac{1}{2^{t-1}}, v'_t = 1$. Note that v'_g differs from v_g as v'_g is for the whole range where f_j is no larger than the upper-bound y_g and v_g only contributes to the range between two consecutive upper-bounds y_{g-1} and y_g. Following the above example, when having not enough knowledge about the outcomes, we can only estimate y_1, y_2, and y_3 with $v'_1 = \frac{1}{2}$, $v'_2 = \frac{3}{4}$ and $v'_3 = 1$. The setting of v'_2 (resp. v'_3), i.e., $v'_2 = v_1 + v_2$ (resp. $v'_3 = v_1 + v_2 + v_3$) is from the fact that y_2 (y_3) can bound all outcomes in groups 1 and 2 (1, 2 and 3), where we can estimate y_g for the corresponding probability with concentration bounds. Intuitively, given a probability v'_3 of success larger than v'_2, the corresponding derived bound y_3 becomes looser compared to y_2 when using fixed sampled records. We will use this intuition to derive bound y_g of $f_j = \|\mathbf{F}_j - \mathbf{G}_j\|_\infty$ and will focus on each dimension of the distribution.

Deriving the upper-bound from each dimension. Recap that F_{ji} is the probability that the record within $\mathcal{S}_j$ for $j = 1, 2$ has the value belonging to the subset $\{e_1, e_2, \ldots, e_i\}$ of all distinct values, on attribute α and G_{ji} is the corresponding probability from the sampling. If $\|\mathbf{F}_j - \mathbf{G}_j\|_\infty$ is no larger than an upper bound y_g, then all $|F_{ji} - G_{ji}|$ should be no larger than y_g for each dimension $i \in [1, s_\alpha]$ by the definition of infinity norm. With union bound, we have $\mathbb{P}(\|\mathbf{F}_j - \mathbf{G}_j\|_\infty \le y_g) \ge 1 - \sum_{i \in [1, s_\alpha]} \mathbb{P}(|F_{ji} - G_{ji}| > y_g)$. Consider each dimension of the distribution. Once we can bound each $|F_{ji} - G_{ji}|$ for $i \in [1, s_\alpha]$ with probabilistic guarantees, the upper-bound y_g of $\|\mathbf{F}_j - \mathbf{G}_j\|_\infty$ can also be derived for a given probability v'_g. Hence, we first derive upper bound y_{gi} of $|F_{ji} - G_{ji}|$ for $i \in [1, s_\alpha]$ with the given success probability using concentration inequalities. The detailed bound is shown as the following lemma.

Lemma 4. *Given attribute α whose support size is s_α, m_j sampled records from subset $\mathcal{S}_j$, failure probability $(1 - v'_g)/s_\alpha$ and parameter $a = \ln(2s_\alpha/(1 - v'_g))$, we have that*

$$|F_{ji} - G_{ji}| \le \max(\epsilon_1, \epsilon_2) = y_{gi}$$

with probability at least $1 - (1 - v'_g)/s_\alpha$, where

$$\epsilon_1 = \frac{2aG_{ji} - \frac{2a}{3} + \sqrt{\left(G_{ji}m_j + \frac{2a}{3}\right)^2 - (2a + m_j)\left(G_{ji}^2 m_j - \frac{2aG_{ji}}{3}\right)}}{2a + m_j},$$

$$\text{and } \epsilon_2 = \frac{a}{m_j} + \sqrt{\frac{2a}{m_j}\left(G_{ji} + \frac{a}{2m_j}\right)}.$$

Focusing on each dimension to derive the bound for a given probability allows for greater flexibility than treating the function holistically. Given the bound y_g from the result of each dimension, we then aggregate by union bound. In particular, with the bounds in Lem. 4 for $|F_{ji} - G_{ji}|$ where $i \in [1, s_\alpha]$, we can bound $\|\mathbf{F}_j - \mathbf{G}_j\|_\infty$ by taking the maximum y_g among each upper-bound y_{gi} of $|F_{ji} - G_{ji}|$, i.e., $y_g = \max_{i \in [1, s_\alpha]} y_{gi}$. Concretely, we have $\mathbb{P}(\|\mathbf{F}_j - \mathbf{G}_j\|_\infty \le y_g) \ge 1 - \sum_{i \in [1, s_\alpha]} \mathbb{P}(|F_{ji} - G_{ji}| > y_g) \ge 1 - \sum_{i \in [1, s_\alpha]} \mathbb{P}(|F_{ji} - G_{ji}| > y_{gi}) = 1 - \sum_{i \in [1, s_\alpha]} (1 - v'_g)/s_\alpha = v'_g$, meaning that given the probability v'_g of success, $f_j = \|\mathbf{F}_j - \mathbf{G}_j\|_\infty$ is upper bounded by y_g.

After that, we discuss how to derive the upper-bound of $\mathbb{E}[f_j]$ based on the above results. With the results of y_g's, we have $\mathbb{E}[f_j] \le \sum_{g=1}^{t} y_g v_g = \sum_{g=1}^{t} y_g (v'_g - v'_{g-1}) = \overline{\mathbb{E}}[f_j]$. The algorithm for upper bounding $\mathbb{E}[f_j]$ where $j = 1, 2$ is formally presented in Algo. 1. Initially, we define the total number t of groups as $\lceil \log_2 n_j \rceil + 1$ and set a series of probabilities of success as shown in Algo. 1 Line 1. In the calculation of each group, we derive the bound y_{gi} for each dimension with Lem. 4 and aggregate them as y_g with the maximum operator (Algo. 1 Lines 2-5). For the last probability v'_t of success, we set the corresponding upper-bound as the largest possible value of f_j, i.e., 1 (Algo. 1 Line 6). Finally, we obtain a bound for $\mathbb{E}[f_j]$, i.e., $\overline{\mathbb{E}}[f_j] = \sum_{g=1}^{t} y_g v_g$ (Algo. 1 Line 8).

The technique above bounds the expectation $\mathbb{E}[f_j]$ of the sampling error, i.e., $f_j^{KS} = \|\mathbf{F}_j - \mathbf{G}_j\|_\infty$, which is used to calculate the difference between the exact metric function value and its estimated result. With the bounds $\overline{\mathbb{E}}[f_j]$ for $j = 1, 2$, we can derive the lower- and upper-bounds of KS-test as shown in the previous subsection.

3.3 Distribution of Failure Probabilities

According to Lem. 3, the confidence radius r_{KS} is the summation of two parts, one for $j = 1$ and the other for $j = 2$, and each part contains the failure probability. A direct question is raised about the feasibility to distribute the failure probability for each part to minimize the confidence radius. In fact, TopKAttr distributes

failure probabilities in a heuristic way. Specifically, they distribute $p'_f\sqrt{h_1}/(\sqrt{h_1}+\sqrt{h_2})$ for $j=1$ and $p'_f\sqrt{h_2}/(\sqrt{h_1}+\sqrt{h_2})$ for $j=2$ without any theoretical analysis, where the total failure probability to calculate confidence radius is p'_f. In the following, we will show how to distribute failure probabilities properly with theoretical results. Removing elements that will not influence the optimization result, this problem can be formalized as

$$\min_{\lambda\in[0,1]} h_1\sqrt{2\ln\left(1/p'_f/\lambda\right)}+h_2\sqrt{2\ln\left(1/p'_f/(1-\lambda)\right)}.$$

When defining $\delta(\lambda)=h_1\sqrt{2\ln(1/p'_f/\lambda)}+h_2\sqrt{2\ln(1/p'_f/(1-\lambda))}$, the target is to find a $\lambda\in[0,1]$ to minimize function $\delta(\lambda)$. The first derivative of $\delta(\lambda)$ is $\delta'(\lambda)=h_2/(1-\lambda)/\sqrt{2\ln(1/p'_f/(1-\lambda))}-h_1/\lambda/\sqrt{2\ln(1/p'_f/\lambda)}$. It is easy to verify that the second derivative $\delta''(\lambda)>0$ for $\lambda\in[0,1]$, which means that $\delta'(\lambda)$ is monotonically increasing in this range. We then need to find the zero of $\delta'(\lambda)$. It is nontrivial to find such a solution in a closed form. Alternatively, as $1-\lambda$ (resp. λ) under the square root with logarithmic function cannot dominate the value of $\delta'(\lambda)$ compared to the same expression outside the square root in practice, we solve $h_2/(1-\lambda)/\sqrt{2\ln(1/p'_f)}-h_1/\lambda/\sqrt{2\ln(1/p'_f)}=0$ instead, which provides a closed-form solution $\lambda=h_1/(h_1+h_2)$. Thus, we distribute failure probabilities $p'_f\cdot h_1/(h_1+h_2)$ for $j=1$ and $p'_f\cdot h_2/(h_1+h_2)$ for $j=2$ in Lem. 3 with theoretical guarantees.

3.4 Extension to other Metric Functions

The analysis in previous subsections focuses on approximating KS-test. As mentioned earlier, our framework estimates the results on each dimension and aggregates them to approximate the real metric function. This technique has the potential to be applied to most metric functions for comparing distributions. In this subsection, we will apply the proposed technique to other metric functions, such as the Chebyshev distance, Earth Mover distance, and Euclidean distance. Instead of estimating the empirical cumulative distribution function of each dimension, we focus on the distribution function according to their definitions and then aggregate them, demonstrating the generality and flexibility of our framework.

Approximation of Chebyshev distance. The analysis of Chebyshev distance is similar to that of KS-test by replacing the empirical cumulative distribution with the data distribution on the corresponding attribute. Following the analysis of KS-test, the absolute difference between the exact Chebyshev distance $D_{CH}(\alpha)$ and its estimation $\hat{D}_{CH}(\alpha)$ on attribute α is $|D_{CH}(\alpha)-\hat{D}_{CH}(\alpha)|\le f_1^{CH}+f_2^{CH}$, where f_j^{CH} is defined as $f_j^{CH}=\|\mathbf{p}_j-\mathbf{q}_j\|_\infty$ for $j=1,2$ with data distributions $\mathbf{p}_j$ and $\mathbf{q}_j$ rather than empirical cummulative distributions $\mathbf{F}_j$ and $\mathbf{G}_j$ in KS-test. Notice that each p_{ji} in $\mathbf{p}_j$ represents the probability of the record in subset $\mathcal{S}_j$ containing value equal to e_i on attribute α and q_{ji} is the corresponding probability among m_j samples from subset $\mathcal{S}_j$. To bound f_j^{CH}, we first derive an upper-bound $\overline{\mathbb{E}}[f_j^{CH}]$ of its expectation $\mathbb{E}[f_j^{CH}]$ by invoking a variant of Algo. 1. The only difference is that we calculate the upper-bound of $|p_{ji}-q_{ji}|$ instead of $|F_{ji}-G_{ji}|$ by the following lemma with given probability v'_g of success in Line 4.

Lemma 5. *Given attribute α with support size s_α, m_j sampled records from $\mathcal{S}_j$, and failure probability $(1-v'_g)/s_\alpha$, $|p_{ji}-q_{ji}|\le\max(\epsilon_1,\epsilon_2)=y_{gi}$ holds with probability at least $1-(1-v'_g)/s_\alpha$ where*

$$\epsilon_1=\frac{2aq_{ji}-\frac{2a}{3}+\sqrt{\left(q_{ji}m_j+\frac{2a}{3}\right)^2-(2a+m_j)\left(q_{ji}^2m_j-\frac{2aq_{ji}}{3}\right)}}{2a+m_j},$$

$$\text{and } \epsilon_2=\frac{a}{m_j}+\sqrt{\frac{2a}{m_j}\left(q_{ji}+\frac{a}{2m_j}\right)}.$$

In Lem. 5, the same parameters as Lem. 4 are omitted due to the limit of space. Besides, the maximum y_g among each upper-bound y_{gi} of $|p_{ji}-q_{ji}|$ bounds f_j^{CH} with probability v'_g of success since $\mathbb{P}(\|\mathbf{p}_j-\mathbf{q}_j\|_\infty\le y_g)\ge 1-\sum_{i\in[1,s_\alpha]}\mathbb{P}(|p_{ji}-q_{ji}|>y_g)\ge 1-\sum_{i\in[1,s_\alpha]}\mathbb{P}(|p_{ji}-q_{ji}|>y_{gi})=1-\sum_{i\in[1,s_\alpha]}(1-v'_g)/s_\alpha=v'_g$ with union bound. Then the lower- and upper-bounds of Chebyshev distance follow Lem. 3 with the substitution of $\overline{\mathbb{E}}[f_j^{KS}]$ with $\overline{\mathbb{E}}[f_j^{CH}]$ calling of the variant of Algo. 1 mentioned above.

Extension to Earth Mover and Euclidean distance. To compute Earth Mover distance, we analyze the gap between the distribution and its estimation in each dimension. Yet, instead of combining the results from each dimension by taking the maximum, we sum up the differences to obtain the gap between the true Earth Mover distance and its estimated value using the union bound, which is derived from the definition of Earth Mover distance. Besides, we adaptively modify the aggregation method for Euclidean distance. It is worth noting that for Earth Mover distance, the absolute difference between the actual and estimated values $|D_{EM}(\alpha)-\hat{D}_{EM}(\alpha)|$ is bounded by $\frac{1}{2}(f_1^{EM}+f_2^{EM})$. As a similar analysis of Euclidean distance, the absolute difference between the exact Euclidean distance $D_{EU}(\alpha)$ and its estimation $\hat{D}_{EU}(\alpha)$ on attribute α has an upper-bound $\frac{1}{\sqrt{2}}(f_1^{EU}+f_2^{EU})$ when defining $f_j^{EU}=\|\mathbf{p}_j-\mathbf{q}_j\|_2$ where $j=1,2$. Rather than deriving an upper-bound $\overline{\mathbb{E}}[f_j^{EM}]$ (resp. $\overline{\mathbb{E}}[f_j^{EU}]$) of $\mathbb{E}[f_j^{EM}]$ (resp. $\mathbb{E}[f_j^{EU}]$) in a closed form to calculate the confidence radius of Earth Mover distance (resp. Euclidean distance) on attribute α as discussed in TopKAttr, we use another variant of Algo. 1 to get $\overline{\mathbb{E}}[f_j^{EM}]$ (resp. $\overline{\mathbb{E}}[f_j^{EU}]$). Specifically, we revise Lines 4-5 in Algo. 1 as calculating the upper-bound y_{gi} of $|p_{ji}-q_{ji}|$ by Lem. 5 with given probability v'_g of success and acquire y_g using $\sum_{i\in[1,s_\alpha]}y_{gi}$ (resp. $\sqrt{\sum_{i\in[1,s_\alpha]}y_{gi}^2}$) to calculate $\overline{\mathbb{E}}[f_j^{EM}]$ (resp. $\overline{\mathbb{E}}[f_j^{EU}]$). During this process, y_g is able to bound f_j^{EM} (resp. f_j^{EU}) with probability v'_g of success from the fact that $\mathbb{P}(\|\mathbf{p}_j-\mathbf{q}_j\|_1\le y_g)\ge 1-\sum_{i\in[1,s_\alpha]}\mathbb{P}(|p_{ji}-q_{ji}|>y_{gi})=1-\sum_{i\in[1,s_\alpha]}(1-v'_g)/s_\alpha=v'_g$ (resp. $\mathbb{P}(\|\mathbf{p}_j-\mathbf{q}_j\|_2\le y_g)\ge 1-\sum_{i\in[1,s_\alpha]}\mathbb{P}(|p_{ji}-q_{ji}|>y_{gi})=1-\sum_{i\in[1,s_\alpha]}(1-v'_g)/s_\alpha=v'_g$) using the union bound. Then, the lower- and upper-bounds of Earth Mover distance (resp. Euclidean distance) can be derived with Lem. 3 by replacing $\overline{\mathbb{E}}[f_j^{KS}]$ with $\overline{\mathbb{E}}[f_j^{EM}]$ (resp. $\overline{\mathbb{E}}[f_j^{EU}]$). As our technique emphasizes each dimension of the distribution, it is flexible to derive bounds for complicated expressions and has potential for other metrics. We will explore this in our future work.

Algorithm 2: Approximate Top-k Attributes

Input: Table $\mathcal{T}$, predicates P_1 and P_2, k, p_f, ϵ
Output: An approximate top-k query answer

1 $C \leftarrow A$, $m \leftarrow m_0$, $R \leftarrow \emptyset$, $i_{\max} \leftarrow \lceil \log_2 \frac{n}{m_0} \rceil$, $p'_f \leftarrow \frac{p_f}{d \cdot i_{\max}}$;
2 **while** $m < n$ **do**
3 Select records satisfying P_1 and P_2 from m records;
4 **for** $\alpha \in C$ **do**
5 Calculate $\hat{D}(\alpha)$, $\underline{D}(\alpha)$ and $\overline{D}(\alpha)$ by Lem. 3;
6 $R \leftarrow$ top-k attributes $\{\alpha'_1, \alpha'_2, \ldots, \alpha'_k\}$ from C according to $\hat{D}(\alpha)$;
7 Find top-k lower-bounds $\{\underline{D}(\alpha^l_1), \underline{D}(\alpha^l_2), \ldots, \underline{D}(\alpha^l_k)\}$ and upper-bounds $\{\overline{D}(\alpha^u_1), \overline{D}(\alpha^u_2), \ldots, \overline{D}(\alpha^u_k)\}$
8 **if** $\overline{D}(\alpha'_i) - \underline{D}(\alpha'_i) \le 2\epsilon$, $\hat{D}(\alpha'_i) - \underline{D}(\alpha^l_i) \le \epsilon$ *and* $\overline{D}(\alpha^u_i) - \hat{D}(\alpha'_i) \le \epsilon$ *for* $i \in [1, k]$ **then**
9 **return** R;
10 **else**
11 $m \leftarrow 2m$;
12 **for** $\alpha \in C$ **do**
13 **if** $\overline{D}(\alpha) < \underline{D}(\alpha^l_k)$ **then**
14 $C \leftarrow C \setminus \{\alpha\}$;
15 $R \leftarrow$ top-k attributes from C according to $D(\alpha)$;
16 **return** R;

4 APPROXIMATE TOP-k QUERY PROCESSING

With the approximations of metric functions, we next show how the proposed framework answers approximate top-k queries.

Approximate top-k algorithm. Algo. 2 presents the pseudo-code for finding approximate top-k attributes with respect to a general metric function. To begin with, all attributes $\alpha \in A$ are included in a candidate set C and the number m of records to retrieve is initialized as m_0. During the iteration, we select m_1 (resp. m_2) records satisfying predicate P_1 (resp. P_2) from m records for the later estimation of metric functions (Algo. 2 Line 3). Then, the estimate, lower- and upper-bounds of the metric function value are calculated for each attribute $\alpha \in C$, denoted as $\hat{D}(\alpha)$, $\underline{D}(\alpha)$ and $\overline{D}(\alpha)$, respectively (Algo. 2 Lines 4-5). Based on their estimates $\hat{D}(\alpha)$, the estimated top-k attributes $\{\alpha'_1, \alpha'_2, \ldots, \alpha'_k\}$ are placed into a result set R. Furthermore, based on the calculated results, we compute the k largest lower-bounds $\{\underline{D}(\alpha^l_1), \underline{D}(\alpha^l_2), \ldots, \underline{D}(\alpha^l_k)\}$ and upper-bounds $\{\overline{D}(\alpha^u_1), \overline{D}(\alpha^u_2), \ldots, \overline{D}(\alpha^u_k)\}$. If the difference between the lower- and upper-bounds of each attribute $\alpha \in R$ is no larger than 2ϵ, and the distances from the estimate $\hat{D}(\alpha)$ to both $\underline{D}(\alpha^l_i)$ and $\overline{D}(\alpha^u_i)$ are no greater than ϵ, then we return set R for the query as the returned k attributes will satisfy the conditions of the approximate top-k query in Def. 2 (Algo. 2 Lines 8-9). Otherwise, the number of retrieved records is doubled to obtain more accurate results. Additionally, we safely prune the attributes whose upper-bounds are smaller than the k-th largest lower-bound to accelerate the query, as they cannot be returned with high probability. If the approximate top-k attributes cannot be found after retrieving n records, the exact metric function values are calculated for each attribute $\alpha \in C$ and the exact top-k results are returned.

Theoretical analysis. Next, we show the correctness of Algo. 2. Specifically, the attributes returned by the algorithm will satisfy the definition of the approximate top-k query with high probability.

Theorem 1. *Algo. 2 returns a result set $R = \{\alpha'_1, \alpha'_2, \ldots, \alpha'_k\}$ of attributes selected from a set A of all attributes, where these k attributes satisfy the approximate top-k query definition in Def. 2 with a probability of at least $1 - p_f$.*

We then discuss the expected running time of Algo. 2.

Theorem 2. *The expected running time of Algo. 2 to answer the approximate top-k query for attribute recommendation is*

$$O\left(\min\left\{dn, \frac{\sum_{i=1}^{d}\left(\log^{\frac{1}{2}}\left(\frac{d\log n}{p_f}\right) + \sum_{j=1}^{2}\log^{\frac{1}{2}}\left(n_j s_{\alpha_i}\right)\right)^2}{\epsilon^2 \eta}\right\}\right).$$

5 THEORETICAL ANALYSIS

We next show the omitted proofs in Sec. 3 and Sec. 4.

Proof of Lem. 3. Recall that $|f_j(\Pi_j) - f_j(\Pi^{rs}_j)| < 1/m_j$ for subset $\mathcal{S}_j$ where m_j is the number of samples from S_j, Π_j is the permutation of records in S_j and Π^{rs}_j is the corresponding permutation when swapping π_r and π_s in Π_j where $s \in [1, m_j]$ and $r \in [m_j + 1, n_j]$. From Lem. 2 of [15], there exists

$$\mathbb{P}\left(f^{KS}_j - \overline{\mathbb{E}}[f^{KS}_j] \ge \gamma_j\right) \le \mathbb{P}\left(f^{KS}_j - \mathbb{E}[f^{KS}_j] \ge \gamma_j\right)$$
$$\le \exp\left(-\frac{2m_j\gamma_j^2(n_j - 1/2)}{n_j - m_j}\left(1 - \frac{1}{2\max(m_j, n_j - m_j)}\right)\right).$$

Setting γ_j as $h_j\sqrt{2\ln((h_1+h_2)/p'_f/h_j)}$ and recalling that h_j is defined as $\sqrt{(n_j - m_j)/4/(n_j - \frac{1}{2})/m_j/(1 - \frac{1}{2\max(m_j, n_j - m_j)})}$, we have $\mathbb{P}(f^{KS}_j \ge \overline{\mathbb{E}}[f^{KS}_j] + \gamma_j) \le p'_f h_j/(h_1 + h_2)$. Since $|D_{KS}(\alpha) - \hat{D}_{KS}(\alpha)| \le f^{KS}_1 + f^{KS}_2$, with union bound,

$$\mathbb{P}\left(\left|D_{KS} - \hat{D}_{KS}\right| \ge \sum_{j=1}^{2}\left(\overline{\mathbb{E}}[f^{KS}_j] + h_j\sqrt{2\ln\frac{h_1 + h_2}{p'_f \cdot h_j}}\right)\right) \le p'_f.$$

By defining $r_{KS} = \sum_{j=1}^{2}(\overline{\mathbb{E}}[f^{KS}_j] + h_j\sqrt{2\ln((h_1+h_2)/p'_f/h_j)})$, we can obtain the result stated in Lem. 3. □

Proof of Lem. 4. In this lemma, we establish an upper-bound on the absolute difference between F_{ji} and its estimate G_{ji} with a high probability. Recall that $F_{ji} = \sum_{k=1}^{i} p_{ji}$ is the empirical cumulative distribution function of attribute α in subset $\mathcal{S}_j$ representing the ratio of attribute values on α that are no larger than e_i for records in subset $\mathcal{S}_j$ where $i \in [1, s_\alpha]$ and $j = 1, 2$. Each sample can be regarded as a Bernoulli trial without replacement. Therefore, we have $F_{ji} = \mathbb{E}[G_{ji}]$. We first examine the probability that the value of $G_{ji} - F_{ji}$ is at least ϵ_1, and we obtain the following inequality

where $a = \ln(2s_\alpha/(1-v'_{k'}))$

$$\mathbb{P}(G_{ji} - F_{ji} \geq \epsilon_1) = \mathbb{P}(F_{ji} \leq G_{ji} - \epsilon_1)$$

$$= \mathbb{P}\left(F_{ji} \leq \frac{G_{ji} + \frac{2a}{3m_j} - \sqrt{\left(G_{ji} + \frac{2a}{3m_j}\right)^2 - \left(\frac{2a}{m_j} + 1\right)\left(G_{ji}^2 - \frac{2aG_{ji}}{3m_j}\right)}}{\frac{2a}{m_j} + 1}\right)$$

$$\leq \mathbb{P}\left(\left(\frac{2a}{m_j} + 1\right)F_{ji}^2 - \left(2G_{ji} + \frac{4a}{3m_j}\right)F_{ji} + G_{ji}^2 - \frac{2aG_{ji}}{3m_j} \geq 0\right)$$

$$= \mathbb{P}\left(G_{ji} \geq F_{ji} + \frac{a}{3m_j} + \sqrt{\frac{a^2}{9m_j^2} + \frac{2aF_{ji}(1-F_{ji})}{m_j}}\right).$$

The last equation holds as the probability of the other side is 0 conditioned on $G_{ji} \geq F_{ji}$. Define δ_1 as $\frac{a}{3m_j} + \sqrt{\frac{a^2}{9m_j^2} + \frac{2aF_{ji}(1-F_{ji})}{m_j}}$, $\delta_1^2 = 2aF_{ji}(1-F_{ji})/m_j + 2a\delta_1/3/m_j$. By [4] (Prop. 1.4), we have

$$\mathbb{P}\left(G_{ji} \geq F_{ji} + \delta_1\right) \leq \exp\left(-\frac{m_j\delta_1^2}{2F_{ij}(1-F_{ji}) + \frac{2\delta_1}{3}}\right) = \frac{1-v'_{k'}}{2s_\alpha}.$$

We then consider the case where $F_{ji} - G_{ji}$ is no smaller than ϵ_2.

$$\mathbb{P}(F_{ji} - G_{ji} \geq \epsilon_2) = \mathbb{P}(F_{ji} \geq G_{ji} + \epsilon_2)$$

$$= \mathbb{P}\left(F_{ji} \geq G_{ji} + \frac{a}{m_j} + \sqrt{\frac{2a}{m_j}\left(G_{ji} + \frac{a}{2m_j}\right)}\right)$$

$$= \mathbb{P}\left(\sqrt{F_{ji}} - \sqrt{\frac{a}{2m_j}} \geq \sqrt{G_{ji} + \frac{a}{2m_j}}\right) \leq \mathbb{P}\left(G_{ji} \leq F_{ji} - \sqrt{\frac{2aF_{ji}}{m_j}}\right).$$

Define δ_2 as $\sqrt{\frac{2aF_{ji}}{m_j}}$. From [10] (Thm. 7), we have

$$\mathbb{P}\left(G_{ji} \leq F_{ji} - \delta_2\right) \leq \exp\left(-\frac{m_j\delta_2^2}{2E[G_{ji}^2]}\right) \leq \exp\left(-\frac{m_j\delta_2^2}{2F_{ji}}\right) = \frac{1-v'_{k'}}{2s_\alpha}.$$

The last inequality comes from the fact that $\mathbb{E}[G_{ji}^2] \leq \mathbb{E}[G_{ji}] = F_{ji}$. With union bound, we have $|F_{ji} - G_{ji}| \leq \max(\epsilon_1, \epsilon_2)$ with probability at least $1 - (1-v'_{k'})/s_\alpha$. □

Proof of Thm. 1. In Algo. 2, the returned attributes based on the approximate results have the properties that $\overline{D}(\alpha'_i) - \underline{D}(\alpha'_i) \leq 2\epsilon$ for $i \in [1,k]$. We define the estimation $\hat{D}(\alpha)$ of $D(\alpha)$ as $\hat{D}(\alpha) = (\underline{D}(\alpha) + \overline{D}(\alpha))/2$. Since $\underline{D}(\alpha'_i) \leq D(\alpha'_i) \leq \overline{D}(\alpha'_i)$, the absolute difference between $D(\alpha)$ and $\hat{D}(\alpha)$ is no larger than ϵ, i.e., $|D(\alpha'_i) - \hat{D}(\alpha'_i)| \leq \epsilon$, which satisfies the first condition in Def. 2.

For attribute α_i^*, where $i \in [1,k]$ and which is the attribute with the i-th largest metric function value, we have $\underline{D}(\alpha_i^l) \leq D(\alpha_i^*) \leq \overline{D}(\alpha_i^u)$ since α_i^l is the attribute with the i-th largest lower-bound and α_i^u is the attribute with the i-th largest upper-bound. The distances between $\hat{D}(\alpha'_i)$ and the boundaries of $D(\alpha_i^*)$, i.e., $\underline{D}(\alpha_i^l)$ and $\overline{D}(\alpha_i^u)$, are all no larger than error bound ϵ. Therefore, the absolute difference between the exact i-th largest metric function value $D(\alpha_i^*)$ and the i-th largest estimated value $\hat{D}(\alpha'_i)$ is no larger than ϵ, i.e., $|D(\alpha_i^*) - \hat{D}(\alpha'_i)| \leq \epsilon$, satisfying the second condition of the definition of the approximate top-k query. The above analysis assumes that all $D(\alpha)$ fall within their respective confidence intervals. With at most $i_{\max}$ iterations, d attributes in each iteration and the failure probability of each call to calculate $\underline{D}(\alpha)$ and $\overline{D}(\alpha)$ is at most p'_f, the total failure probability of Algo. 2 is at most $p'_f \cdot d \cdot i_{\max} = p_f$. It is easy to verify that when returning top-k attributes with exact values, all returned attributes satisfy Def. 2. Hence, Algo. 2 returns attributes fulfilling Def. 2 with probability at least $1 - p_f$. □

Proof of Thm. 2. Regarding time complexity, we focus on the cost of data access as computation cost can be ignored compared to the time required to read data from memory.

We first define a larger set U as the union of attributes with top-k estimated metric function values, k largest lower- and upper-bounds, i.e., $U = \{\alpha'_1, \alpha'_2, \ldots, \alpha'_k\} \cup \{\alpha_1^l, \alpha_2^l, \ldots, \alpha_k^l\} \cup \{\alpha_1^u, \alpha_2^u, \ldots, \alpha_k^u\}$. Instead of directly analyzing the termination conditions in Algo. 2, we consider stricter conditions: for any attribute $\alpha \in U$, the difference between its upper-bound $\overline{D}(\alpha)$ and lower-bound $\underline{D}(\alpha)$ is no larger than 2ϵ. Before continuing, we will prove that these conditions are stricter. Since $\overline{D}(\alpha) - \underline{D}(\alpha) \leq 2\epsilon$ for $\alpha \in U$ and $\{\alpha'_1, \alpha'_2, \ldots, \alpha'_k\} \subset U$, the first condition in Algo. 2 Line 8 is satisfied. Consider any $i \in [1,k]$ and $\hat{D}(\alpha) = (\underline{D}(\alpha) + \overline{D}(\alpha))/2$ for $\alpha \in U$. Recall that $\hat{D}(\alpha'_i)$ is the i-th largest estimated metric function value. At least i lower-bounds, i.e., $\{\underline{D}(\alpha'_1), \underline{D}(\alpha'_2), \ldots, \underline{D}(\alpha'_i)\}$, are no smaller than $\hat{D}(\alpha'_i) - \epsilon$ as $\hat{D}(\alpha'_i) - \epsilon = (\underline{D}(\alpha'_i) + \overline{D}(\alpha'_i))/2 - \epsilon \leq \underline{D}(\alpha'_i)$ for $i \in [1,k]$. Since $\underline{D}(\alpha_i^l)$ is the i-th lower-bound, $\underline{D}(\alpha_i^l) \geq \hat{D}(\alpha'_i) - \epsilon$ and the second condition in Algo. 2 Line 8 is also satisfied. As $\overline{D}(\alpha_i^u)$ is the i-th largest upper-bound and $\hat{D}(\alpha_i^u) = (\underline{D}(\alpha_i^u) + \overline{D}(\alpha_i^u))/2 \geq \overline{D}(\alpha_i^u) - \epsilon$, at least i estimated values, i.e., $\{\hat{D}(\alpha_1^u), \hat{D}(\alpha_2^u), \ldots, \hat{D}(\alpha_i^u)\}$, are no smaller than $\overline{D}(\alpha_i^u) - \epsilon$. Since α'_i is the attribute with the i-th estimated metric function value, $\hat{D}(\alpha'_i) \geq \overline{D}(\alpha_i^u) - \epsilon$ and the third condition in Algo. 2 Line 8 is then satisfied. Hence, when $\overline{D}(\alpha) - \underline{D}(\alpha) \leq 2\epsilon$ for all $\alpha \in U$, Algo. 2 will terminate.

We next analyze the sample size for these stricter conditions. When the difference between the upper-bound $\overline{D}(\alpha)$ and lower-bound $\underline{D}(\alpha)$ is no larger than 2ϵ for $\alpha \in U$, the confidence radius r_{KS} is no larger than error bound ϵ. To simplify the analysis, we distribute the failure probabilities equally for $j = 1$ and $j = 2$ and the requirement is that $\sum_{j=1}^{2}(\overline{E}[f_j] + h_j\sqrt{2\ln(2/p'_f)}) \leq \epsilon$. Recall that $h_j = \sqrt{(n_j - m_j)/4/(n_j - 1/2)/m_j/(1 - 1/\max(m_j, n_j - m_j))}$ for $j = 1, 2$. Since $\max(m_j, n_j - m_j) \geq n_j/2$, the algorithm terminates when $\sum_{j=1}^{2}(\overline{E}[f_j] + \sqrt{n_j\ln(2/p'_f)/2/m_j/(n_j - 5/2)}) \leq \epsilon$. We then discuss ϵ_1 and ϵ_2 for $\overline{E}[f_j]$ where $j = 1, 2$. For ϵ_1, we have

$$\epsilon_1 \leq \frac{2aG_{ji}}{m_1} - \frac{2a}{3m_j} + \sqrt{\frac{2aG_{ji}(1-G_{ji})}{m_j} + \frac{4a^2(1+3G_{ji})}{9m_j^2}}$$

$$\leq \frac{2aG_{ji}}{m_j} + \sqrt{\frac{a}{2m_j}} + \frac{2a}{3m_j}\sqrt{3G_{ji}} = \frac{2a}{m_j}\left(G_{ji} + \sqrt{\frac{G_{ji}}{3}}\right) + \sqrt{\frac{a}{2m_j}}.$$

As for ϵ_2, we have $\epsilon_2 \leq 2a/m_j + \sqrt{2aG_{ji}/m_j}$. Therefore, $y_{gi} = \max(\epsilon_1, \epsilon_2) \leq 4a/m_j + \sqrt{2a/m_j}$. As $y_g = \max_{i\in[1,s_\alpha]} y_{gi}$, $\overline{E}[f_j] = \sum_{g=1}^{t} y_g v_g$, and $a \leq \ln(2n_j s_\alpha)$ when considering Algo. 1 and Lem. 4, $\overline{E}[f_j] \leq 4\ln(2n_j s_\alpha)/m_j + \sqrt{2\ln(2n_j s_\alpha)/m_j}$. Denote m' as $\min(m_1, m_2)$.

Table 2: Summary of datasets

Dataset	Rows	Columns
Enem	69,940,536	67
Census American Housing	87,154,886	175
Census American Population	185,760,233	192
Airline Reporting Carrier On-Time	194,385,636	53

Then we have a striter requirement that

$$\sum_{j=1}^{2}\left(\frac{4\ln(2n_j s_\alpha)}{m'}+\sqrt{\frac{2\ln(2n_j s_\alpha)}{m'}}+\sqrt{\frac{n_j\log(2/p'_f)}{2m'(n_j-5/2)}}\right)\le\epsilon,$$

which is equal to the following inequality

$$\epsilon m'-\sum_{j=1}^{2}\left(\left(\sqrt{2\ln(2n_j s_\alpha)}+\sqrt{\frac{n_j\ln(2/p'_f)}{2(n_j-5/2)}}\right)\sqrt{m'}+4\ln(2n_j s_\alpha)\right)\ge 0.$$

Solve the above inequality of m' and we can observe that when the smaller number of sampling records satisfying predicates P_1 or P_2

$$m'\ge m^*=\frac{b^2+8\epsilon\sum_{j=1}^{2}\ln(2n_j s_\alpha)+b\sqrt{b^2+16\epsilon\sum_{j=1}^{2}\ln(2n_j s_\alpha)}}{2\epsilon^2},$$

the termination conditions in Algo. 2 are all satisfied, where we define b as $\sum_{j=1}^{2}(\sqrt{2\ln(2n_j s_\alpha)}+\sqrt{n_j\ln(2/p'_f)/2/(n_j-5/2)})$. During each iteration, we will successively double the number of retrieved records and evaluate whether it has reached a sufficient magnitude to satisfy the termination condition. So when the algorithm terminates using the approximate results, $m'\le 2m^*$ with at least $1-p_f$ probability. According to the setting, $i_{\max}=\log_2(\lceil n/m_0\rceil)$ and $p'_f=p_f/i_{\max}/d$. Then the total number m^* of sampled records satisfying predicates is $O((\log^{\frac{1}{2}}(d\log n/p_f)+\sum_{j=1}^{2}\log^{\frac{1}{2}}(n_j s_\alpha))^2/\epsilon^2)$. There are at most d attributes in the candidate set. Also, the number of records we access is no larger than the total number n of records in table $\mathcal{T}$. Denote η as the smaller selectivity for predicates P_1 and P_2. Thus, the expected running time of Algo. 2 is as follows

$$O\left(\min\left\{dn,\frac{\sum_{i=1}^{d}\left(\log^{\frac{1}{2}}\left(\frac{d\log n}{p_f}\right)+\sum_{j=1}^{2}\log^{\frac{1}{2}}(n_j s_{\alpha_i})\right)^2}{\epsilon^2\eta}\right\}\right).$$ □

Effectiveness on Chebyshev distance for various k. Figure 4 shows that AFFAIR is more than 80× faster than Exact in all cases. Specifically, AFFAIR has an up to 819× speedup over Exact when $k=1$ on dataset pus. At the same time, AFFAIR finds the exact top-k attributes.

6 EXPERIMENTS

This section evaluates the proposed framework against the alternatives experimentally. All experiments are conducted on a Linux machine with an Intel Xeon CPU @2.3GHz and 448GB memory.

6.1 Settings

Datasets. To validate the efficiency and effectiveness of the algorithms, we utilize four large real datasets: Enem, Census American Housing (hus), Census American Population (pus), and Airline Reporting Carrier On-Time (airline). Each dataset consists of more than 60 million records and no fewer than 50 attributes. These large real datasets are publicly available online and were previously tested in [48, 49], with statistics summarized in Tab. 3. To obtain two subsets $\mathcal{S}_1$ and $\mathcal{S}_2$ of a dataset, we randomly generate predicates P_1 and P_2 that satisfy corresponding selectivities. Throughout all experiments, each metric is averaged over 10 cases.

Algorithms. We compare our approximation framework for attribute recommendation (AFFAIR) with the state-of-the-art solution TopKAttr [48] and exact method by scanning all records (dubbed as Exact). For Earth Mover distance and Euclidean distance, we compare our AFFAIR with TopKAttr and Exact. For TopKAttr, we set the number of records in each partition to be 100K, in line with the configuration mentioned in [49] which uses the same methodology as TopKAttr to find top-k attributes but focuses on the single-machine in-memory setting. Only Exact is compared on KS-test and Chebyshev distance since TopKAttr does not provide a solution for these metric functions. The evaluation focuses on the running time for all algorithms and the F1-measure of the results returned by approximation algorithms. All algorithms are implemented with C++ and compiled with full optimization.

Parameters. Both AFFAIR and TopKAttr require a failure probability in the top-k algorithms, denoted by p_f, which is set to $p_f=1/n$ where n is the total number of records in the dataset. In our AFFAIR, an error bound ϵ is provided to trade off the query efficiency and accuracy. As will be discussed in Sec. 6.4, we set the default value of ϵ to 0.05 as it balances efficiency and accuracy well. For the initial sample m_0 (Ref. to Algo. 2), we set it to 1024 to align it to a page size if the record values are integers.

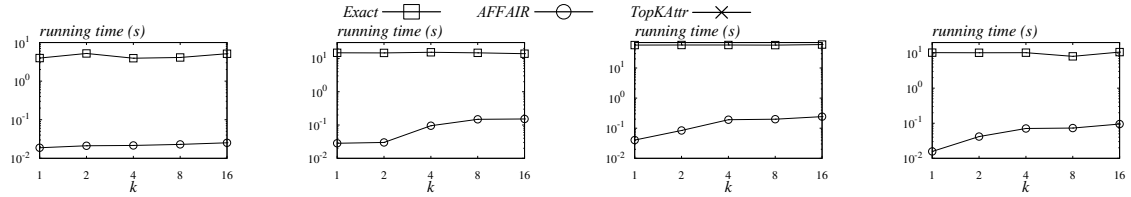
6.2 Top-k Query Processing

In this subsection, we present the evaluation of our framework against the alternatives for performing top-k queries using four different metric functions: KS-test, Chebyshev distance, Earth Mover distance, and Euclidean distance. To assess the influence of parameter k on performance, we vary k from 1 to 16 and report results for $k=\{1,2,4,8,16\}$ while maintaining a fixed error bound of $\epsilon=0.05$ and selectivity of $\eta=0.5$. We then discuss the experimental results for each metric function in detail.

Effectiveness on KS-test and Chebshev distance. Fig. 2 shows the running time of top-k algorithms on the KS-test. For all cases, AFFAIR is 90× faster than Exact. When $k=1$ on dataset pus, AFFAIR achieves an up to 1430× speed-up over Exact. As for accuracy, AFFAIR returns the same top-k attributes as Exact with 100% F1-measure in all cases on all datasets, as shown in Fig. 3. It shows that our AFFAIR gains superb efficiency without any trade-off to accuracy, which is the preferred choice. For remaining metric functions, our framework consistently achieves 100% F1-measure on all datasets in all cases.

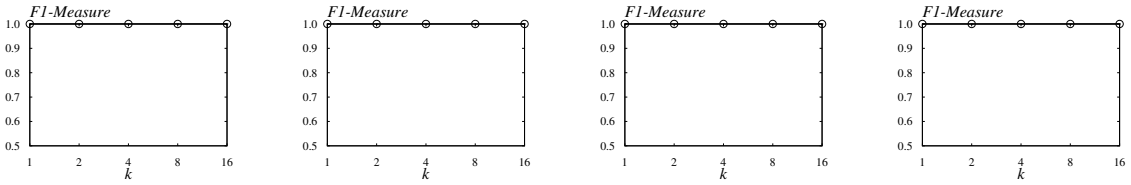
Next, we examine the performance of AFFAIR with Chebshev distance. Fig. 4 shows that AFFAIR is more than 80× faster than Exact in all cases. Specifically, AFFAIR has an up to 819× speed-up over Exact when $k=1$ on dataset pus. Also, AFFAIR achieves high accuracy as the alternatives.

Effectiveness on Earth Mover and Euclidean distance. Fig. 6 demonstrates that AFFAIR outperforms both Exact and TopKAttr in all cases on Earth Mover distance, with up to 67× speed-up over Exact when $k=2$ on dataset pus, and up to 12× speed-up over



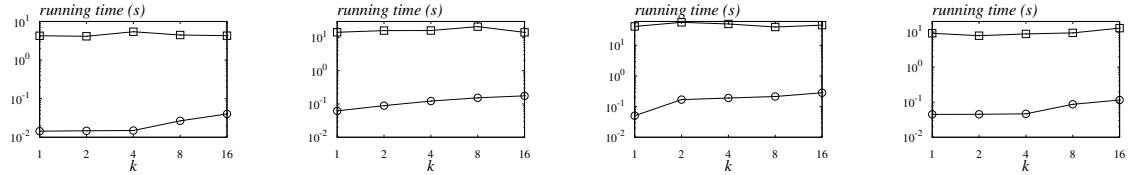
(a) enem (b) hus (c) pus (d) airline

Figure 2: Varying k: Running time of the top-k algorithms on KS-test.



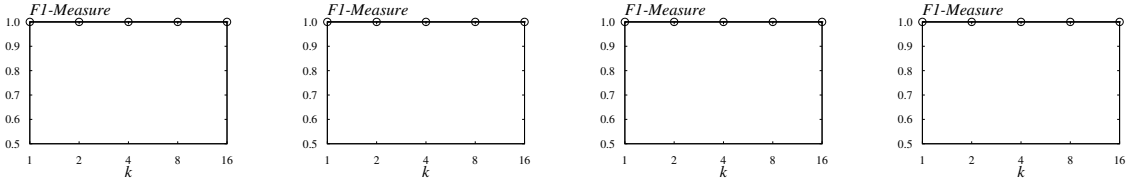
(a) enem (b) hus (c) pus (d) airline

Figure 3: Varying k: F1-Measure of the query result on KS-test.



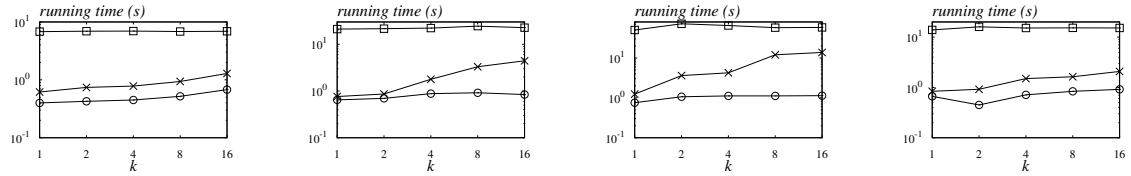
(a) enem (b) hus (c) pus (d) airline

Figure 4: Varying k: Running time of top-k algorithms on Chebyshev distance.



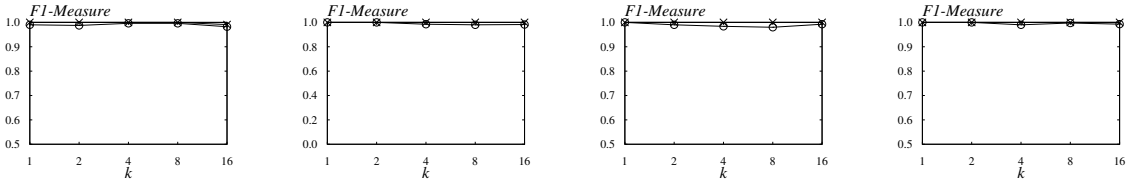
(a) enem (b) hus (c) pus (d) airline

Figure 5: Varying k: Query F1-Measure of top-k algorithms on Chebyshev distance.



(a) enem (b) hus (c) pus (d) airline

Figure 6: Varying k: Running time of top-k algorithms on Earth Mover distance.

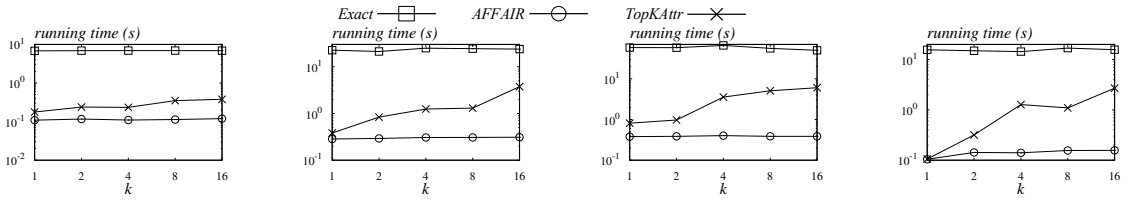


(a) enem (b) hus (c) pus (d) airline

Figure 7: Varying k: Query F1-Measure of top-k algorithms on Earth Mover distance.

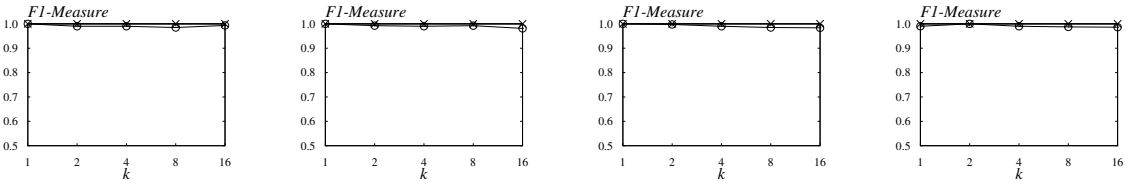
TopKAttr when $k = 16$ on dataset pus. Besides, our AFFAIR returns the attribute set with nearly 100% F1-Measure.

Next, we examine the performance of all methods on Euclidean distance. As shown in Fig. 8, our AFFAIR outperforms alternatives in terms of efficiency in all cases. Notably, when $k = 8$, AFFAIR is up to 165× faster than Exact on pus dataset; our solution further obtains up to 16× speed-up over TopKAttr when $k = 16$ on hus. At



(a) enem (b) hus (c) pus (d) airline

Figure 8: Varying k: Running time of top-k algorithms on Euclidean distance.



(a) enem (b) hus (c) pus (d) airline

Figure 9: Varying k: Query F1-Measure of top-k algorithms on Euclidean distance.

Table 3: Summary of datasets

Dataset	Rows	Columns
Enem	69,940,536	67
Census American Housing	87,154,886	175
Census American Population	185,760,233	192
Airline Reporting Carrier On-Time	194,385,636	53

the same time, AFFAIR finds the top-k attributes with nearly 100% F1-Measure.

6.3 Impact of Selectivity η

We also conduct a set of experiments to evaluate the impact of selectivity η. To do this, we vary η from 0.1 to 1 and show the results when η is $\{0.1, 0.25, 0.5, 0.75\}$. In this set of experiments, error bound ϵ is set to 0.05, and k is fixed at 16.

Effectiveness on KS-test and Chebyshev distance. Fig. 10 shows the running time on KS-test. Across all instances, AFFAIR is at least 50× faster than Exact and achieves up to two orders of magnitude speed-up over Exact on four datasets. Again, AFFAIR is at least 50× faster than Exact in all cases and achieves up to two orders of magnitude speed-up over Exact on four datasets. In terms of accuracy, AFFAIR produces identical results to Exact.

Effectiveness on Earth Mover and Euclidean distance. Fig.14 reports the running time with Earth Mover distance as the metric. It shows that AFFAIR outperforms both Exact and TopKAttr in all cases in terms of running time. When $\eta = 0.75$ on dataset pus, our AFFAIR is up to 75× faster than Exact and exhibits a significant speed-up of up to 20× over TopKAttr. Moreover, AFFAIR demonstrates consistently high accuracy as the alternatives.

Fig. 16 reports the running time with Euclidean distance as the metric. It illustrates that AFFAIR is more than 40× faster than Exact and has a significant speed-up over TopKAttr in all cases based on their respective running time. When $\eta = 0.75$ on dataset pus, AFFAIR is up to 181× faster than Exact, while on dataset hus, AFFAIR achieves an up to 24× speed-up over TopKAttr. Again, AFFAIR returns the attributes with nearly 100% F1-Measure.

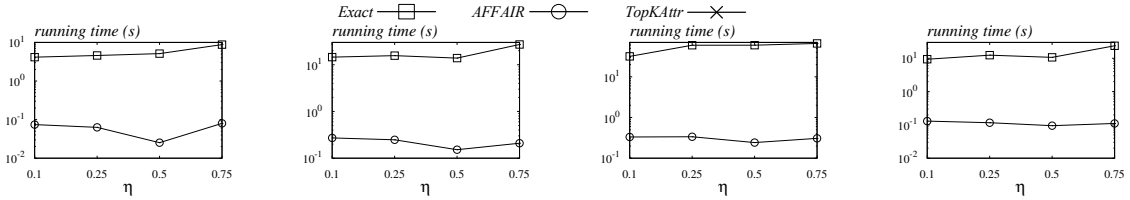
6.4 Tuning Error Bound Parameter ϵ

Recap that our AFFAIR has an error-bound parameter ϵ to balance efficiency and accuracy. Next, we evaluate the impact of ϵ on the top-k queries for four distribution comparison metric functions: KS-test, Chebyshev distance, Earth Mover distance, and Euclidean distance. We vary ϵ from 0.001 to 0.5 and report the results when ϵ is equal to $\{0.001, 0.0025, 0.005, 0.01, 0.025, 0.05, 0.1, 0.25, 0.5\}$ when fixing k at 16 and selectivity η at 0.5. As shown in Figs. 18-21 (a), the running time of our approximate top-k query on KS-test and Chebyshev distance sharply decreases with increasing ϵ. However, for $\epsilon > 0.05$, the F1-measures drop below 98% on KS-test for dataset pus, as shown in Fig. 18 (b), and on Chebyshev distance for datasets enem, hus, and pus, as shown in Fig. 19 (b). When $\epsilon > 0.05$, the F1-measures fall below 98% on Earth Mover distance for datasets hus and pus as shown in Fig. 20 (b) and on Euclidean distance for all four datasets as shown in Fig. 21 (b). Therefore, we use $\epsilon = 0.05$ as the default setting of all four tested metric functions since it achieves the best trade-off between efficiency and accuracy.

6.5 User Study

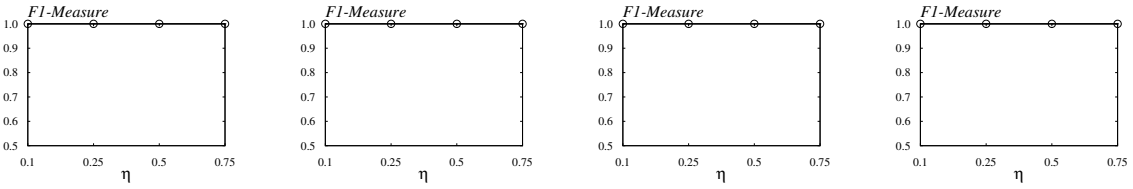
Next, we have a user study to assess the effectiveness of our recommended attributes based on metric functions for real users.

Comparing different metric functions. Following SeeDB [47], we use the pus dataset and the task of analyzing the impact of marital status. Similar to the user study in SeeDB [47], we set the two predicates married="True" and married="False" for trend analysis. The goal is to find the attributes such that under these two different predicates, the two distributions show different trends that can be used for analysis. We first visualize the two distributions for 100 attributes when the predicate married is "True" and "False", respectively. In SeeDB [47], 5 experts are then asked to classify each visualization as interesting or not interesting in the context of the task. Then, they use the majority vote to label the attribute as "interesting" or "not interesting". We follow their setting and invite 5 experts to manually label the distribution deviation for each attribute as interesting and not interesting. We then derive the rank of each attribute by taking the average score of these 5 experts. For instance, if 4 out of 5 experts think the attribute is interesting,



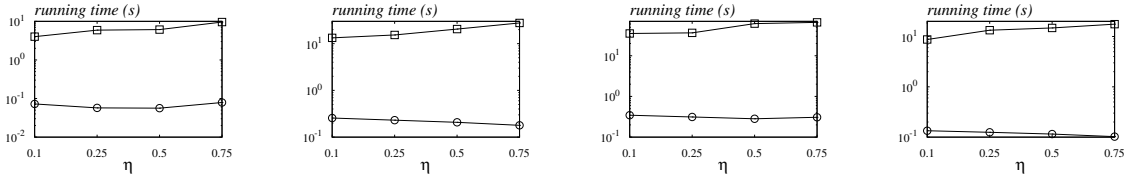
(a) enem (a) hus (c) pus (b) airline

Figure 10: Varying η: Running time on KS-test.



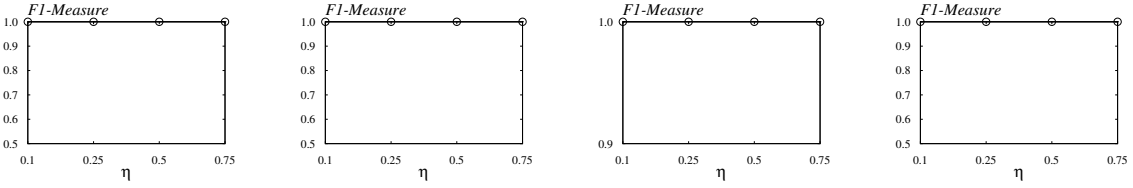
(a) enem (a) hus (c) pus (b) airline

Figure 11: Varying η: F1-Measure of the query result on KS-test.



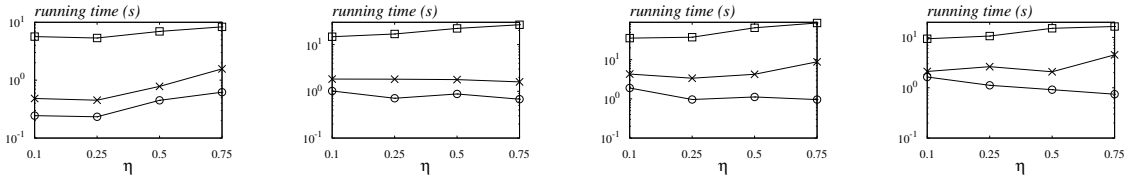
(a) enem (a) hus (c) pus (b) airline

Figure 12: Varying η: Running time on Chebyshev distance.



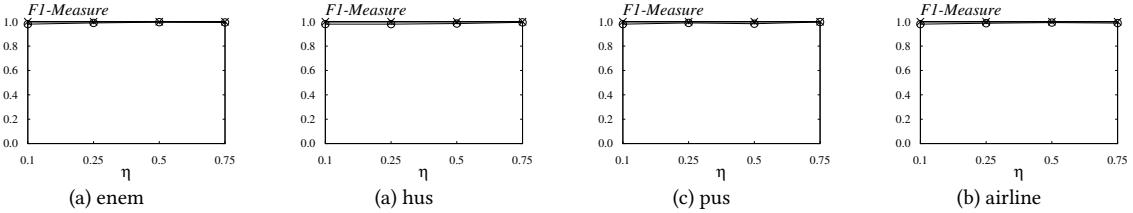
(a) enem (a) hus (c) pus (b) airline

Figure 13: Varying η: F1-Measure of the query result on Chebyshev distance.



(a) enem (b) hus (c) pus (d) airline

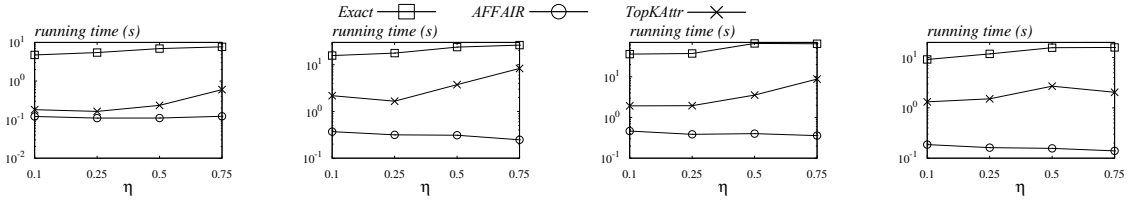
Figure 14: Varying η: Running time on Earth Mover distance.



(a) enem (a) hus (c) pus (b) airline

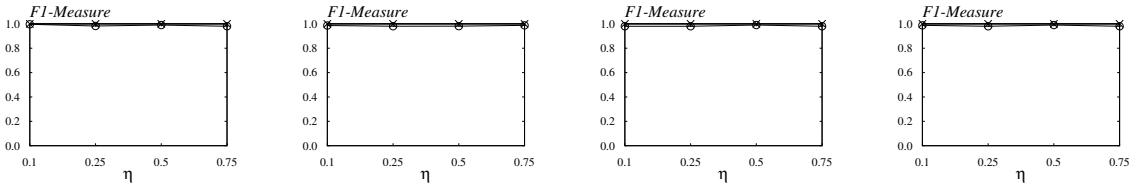
Figure 15: Varying η: F1-Measure of the query result on Earth Mover distance.

the score is 0.8. We then rank the attributes by the scores. When the attributes have the same score, we rank them by the alphabetic order of the attribute name to make them consistent no matter how we change the metric function. Among these 100 attributes, 9 attributes have a score of 1, 9 have a score of 0.8, 9 have a score of 0.6, 6 have a score of 0.4, 6 have a score of 0.2, and all remaining have a score of 0. Thus, 27 attributes are labeled as "interesting", and the remaining attributes are "not interesting". In Fig. 22, each row indicates an attribute and for the same row, the attribute is the same for all metric functions, ranked according to the scores



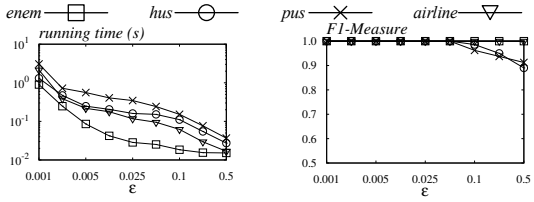
(a) enem (a) hus (c) pus (b) airline

Figure 16: Varying η: Running time on Euclidean distance.



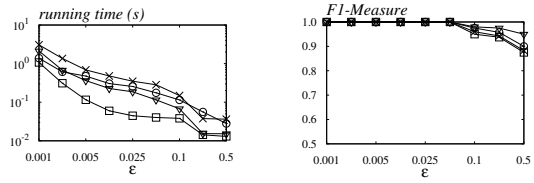
(a) enem (a) hus (c) pus (b) airline

Figure 17: Varying η: F1-Measure of the query result on Euclidean distance.



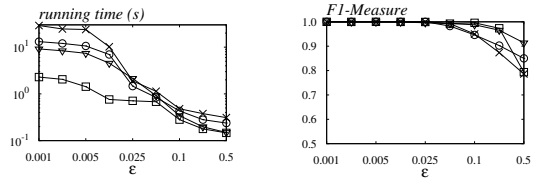
(a) Running Time (b) F1-Measure

Figure 18: Tuning ϵ: KS-test.



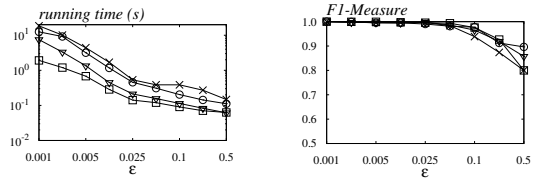
(a) Running Time (b) F1-Measure

Figure 19: Tuning ϵ: Chebyshev distance.



(a) Running Time (b) F1-Measure

Figure 20: Tuning ϵ: Earth Mover distance.



(a) Running Time (b) F1-Measure

Figure 21: Tuning ϵ: Euclidean distance.

of experts. The uppermost attribute has the highest score 1.0, and the attribute at the bottom has the lowest score 0.0 ranked by the experts. Next, a colored heatmap is drawn based on the rank of each attribute by using each metric function.

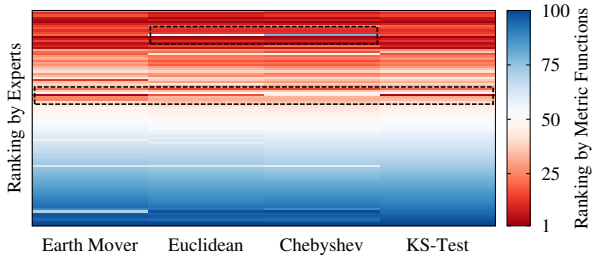


Figure 22: Metric functions ranking.

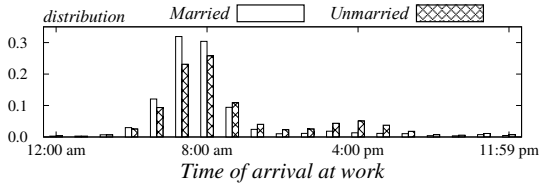


Figure 23: Interesting attribute: high ranking with Earth Mover&KS-test; low ranking with Euclidean&Chebyshev.

The heatmap in Fig. 22 shows a concentration of red bands at the top and blue bands at the bottom, indicating that all four metric functions perform well in identifying interesting trends with high values while filtering out uninteresting ones, which suggests that the attributes recommended by our framework are of high quality.

However, the figure also reveals that there is no one-size-fits-all metric function for the attribute recommendation task. To explain, firstly, as we can observe from the dashed box at the top in Fig. 22, there exists an attribute "Time of arrival at work" that is ranked high by experts but is ranked medium with the Euclidean distance (in white color) and ranked low with Chebyshev distance (in blue color). Fig. 23 shows the detailed distribution of the attribute "Time of arrival at work" for those who are married and not married. There is a significant trend change in the time of arrival at work depending on the marital status, and such a trend can be potentially used in other data analytic tasks. For example, companies can leverage this trend to strategically place TV advertisements targeting specific users on public transportation at different times of the day.

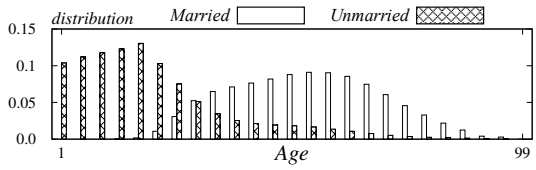


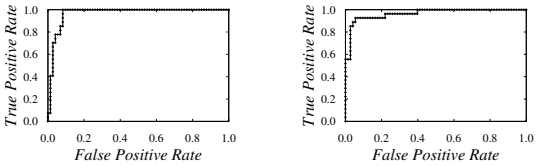
Figure 24: Non-interesting attribute: high ranking with Earth Mover, Euclidean, and KS-test; low ranking with Chebyshev.

Besides, for the dashed box below in Fig. 22, we can find that the attribute "Age", which is "not interesting" with a score of 0.0 by the experts, is ranked high (with deep red) by Earth Mover distance, Euclidean distance, and KS-test. Only Chebyshev provides a low rank to this attribute. Fig. 24 shows the detailed distribution. There is an obvious trend that a young age tends not to be married while adults are more likely to be married. This is expected from common sense and is not interesting from a data scientist's perspective.

The results show that there is no one-size-fits-all metric function for attribute recommendation and it is necessary to include a general framework to support more metric functions.

Effectiveness in attribute recommendation. Additionally, we follow SeeDB and adopt the "receiver operating characteristic" (ROC) curve, a common concept in data mining, to show the relationship between the false positive rate and the true positive rate. Here, in our studied problem, the true positive rate measures the ratio of retrieved interesting attributes over the whole set of interesting attributes. The false positive rate measures the ratio of non-interesting attributes that are falsely returned in the top-k answer over the total number of non-interesting attributes. The goal is to achieve a high true positive rate and a low false positive rate. The area under the ROC curve (also called the AUC), measures how well a top-k algorithm balances the true positive rate and false positive rate. The higher the AUC is, the better the top-k algorithm is. Generally, when AUC is above 0.9, the top-k algorithm is considered to perform excellently. Fig. 25 shows the ROC curve and the corresponding AUC for these four metric functions, where the top-k algorithm with all four metric functions can gain excellent AUC, all above 0.95. Besides, KS-test achieves the best AUC of 0.978, demonstrating a better alternative to Euclidean and Earth Mover distance. We further combine all four metric functions to obtain the medium score and use the score to derive the top-k answer. This strategy gains an AUC of 0.982, achieving a better performance. It shows the potential to improve the effectiveness by supporting different metric functions with a generalized framework.

In summary, our experiments show that all four metric functions effectively recommends interesting attributes for trend analysis. Yet, there is no metric function that is best suited for all scenarios, as indicated by our findings. Providing a general framework to support more metric functions other than Euclidean and Earth Mover distance is indeed important. With more metric functions, we may combine them to avoid the limitations of a specific metric function and have the potential to gain better results.



(a) Earth Mover, $AUC = 0.967$ (b) Euclidean, $AUC = 0.965$

(c) Chebyshev, $AUC = 0.953$ (d) KS-test, $AUC = 0.978$

Figure 25: ROC of the four metric functions.

7 RELATED WORK

There are existing works [38, 39] that identify sub-cubes of a cube having the largest deviation among all sub-cubes, similar to the attribute recommendation problem in [47, 48] which finds attributes with the largest variations for ad hoc queries. These works use data mining techniques such as table analysis method [39] and entropy [38]. Seo et. rank attributes to enable the understanding of distributions and discovery of relationships [40]. Based on the statistics of data, VizDeck recommends visualizations with a series of heuristics [25]. Voyager [55] provides univariate summaries for each attribute, suggests additional attributes beyond the selected one by users, and ranks visualizations according to data properties. SeeDB [47] and TopKAttr [48] recommend attributes with the largest deviations between two subsets of records under ad hoc queries.

In feature selection, existing works can be categorized as filter methods, wrapper methods, and embedded methods [27]. This review focuses mainly on the filter methods that rely on statistical methods to evaluate each attribute. Laplacian score [20] selects features that best preserve the data manifold structure. Information gain [19] measures the importance of a feature from its correlation with class labels. Besides, fast correlation-based filter [56] considers both feature-class correlation and feature-feature correlation. In addition, the Chi-square score [29] leverages the independence test to validate whether the feature is independent of the class label.

Sampling methods are wildly used in databases [33]. Olken et al. design data structures and algorithms for sampling from relational databases [34]. Toivonen adopts sampling to find association rules from large databases efficiently [46]. Additionally, approximate query processing (AQP) is a technique used in database systems to provide quick and approximate answers to complex queries based on samples. Recently, Chaudhuri et al. point out two routes: ceding control over accuracy to the user and leveraging AQP system for exploratory queries, to integrate AQP into data platforms. Later, VerdictDB [35], an AQP framework is proposed to work with all off-the-shelf engines using a middleware architecture.

8 CONCLUSION

This paper proposes a general approximation framework AFFAIR for attribute recommendation that returns k attributes efficiently while providing theoretical guarantees. The framework accommodates a broad range of metric functions such as KS-test, Chebyshev distance, Earth mover distance, Euclidean distance, and has the potential to integrate more metrics. Extensive experiments show that

AFFAIR is an order of magnitude faster than the state-of-the-art approximate solution while maintaining consistently high accuracy.

REFERENCES

[1] James Abello, Panos M Pardalos, and Mauricio GC Resende. 2013. *Handbook of massive data sets*. Vol. 4.

[2] Sanjay Agrawal, Surajit Chaudhuri, Lubor Kollár, Arunprasad P. Marathe, Vivek R. Narasayya, and Manoj Syamala. 2005. Database tuning advisor for microsoft SQL server 2005: demo. In *SIGMOD*. 930–932.

[3] Nikolaus Augsten, Denilson Barbosa, Michael H. Böhlen, and Themis Palpanas. 2010. TASM: Top-k Approximate Subtree Matching. In *ICDE*. 353–364.

[4] Rémi Bardenet and Odalric-Ambrym Maillard. 2015. Concentration inequalities for sampling without replacement. *Bernoulli* 21, 3 (2015), 1361 – 1385.

[5] Peter A. Boncz, Torsten Grust, Maurice van Keulen, Stefan Manegold, Jan Rittinger, and Jens Teubner. 2006. MonetDB/XQuery: a fast XQuery processor powered by a relational engine. In *SIGMOD*. 479–490.

[6] Charles Bontempo and George Zagelow. 1998. The IBM Data Warehouse Architecture. *Commun. ACM* 41, 9 (1998), 38–48.

[7] Lisi Chen and Shuo Shang. 2019. Approximate spatio-temporal top-k publish/subscribe. *WWW* 22, 5 (2019), 2153–2175.

[8] Xingguang Chen and Sibo Wang. 2021. Efficient Approximate Algorithms for Empirical Entropy and Mutual Information. In *SIGMOD*, Guoliang Li, Zhanhuai Li, Stratos Idreos, and Divesh Srivastava (Eds.). 274–286.

[9] Xingguang Chen, Fangyuan Zhang, and Sibo Wang. 2022. Efficient Approximate Algorithms for Empirical Variance with Hashed Block Sampling. In *SIGKDD*. 157–167.

[10] Fan R. K. Chung and Lincoln Lu. 2006. Concentration Inequalities and Martingale Inequalities: A Survey. *Internet Math.* 3, 1 (2006), 79–127.

[11] John Clear, Debbie Dunn, Brad Harvey, Michael L. Heytens, Peter Lohman, Abhay Mehta, Mark Melton, Lars Rohrberg, Ashok Savasere, Robert M. Wehrmeister, and Melody Xu. 1999. NonStop SQL/MX Primitives for Knowledge Discovery. In *SIGKDD*. 425–429.

[12] Benoît Dageville, Thierry Cruanes, Marcin Zukowski, Vadim Antonov, Artin Avanes, Jon Bock, Jonathan Claybaugh, Daniel Engovatov, Martin Hentschel, Jiansheng Huang, Allison W. Lee, Ashish Motivala, Abdul Q. Munir, Steven Pelley, Peter Povinec, Greg Rahn, Spyridon Triantafyllis, and Philipp Unterbrunner. 2016. The Snowflake Elastic Data Warehouse. In *SIGMOD*. 215–226.

[13] Elena Deza, Michel Marie Deza, Michel Marie Deza, and Elena Deza. 2009. *Encyclopedia of distances*.

[14] Ran El-Yaniv and Dmitry Pechyony. 2006. Stable Transductive Learning. In *COLT*, Vol. 4005. 35–49.

[15] Ran El-Yaniv and Dmitry Pechyony. 2009. Transductive Rademacher Complexity and its Applications. *J. Artif. Intell. Res.* 35 (2009), 193–234.

[16] Sonja Engmann and Denis Cousineau. 2011. Comparing distributions: the two-sample anderson-darling test as an alternative to the kolmogorov-smirnoff test. *Journal of applied quantitative methods* 6, 3 (2011).

[17] Charles H Gibson. 2012. *Financial reporting and analysis*.

[18] Anurag Gupta, Deepak Agarwal, Derek Tan, Jakub Kulesza, Rahul Pathak, Stefano Stefani, and Vidhya Srinivasan. 2015. Amazon Redshift and the Case for Simpler Data Warehouses. In *SIGMOD*. 1917–1923.

[19] Isabelle Guyon and André Elisseeff. 2003. An Introduction to Variable and Feature Selection. *J. Mach. Learn. Res.* 3 (2003), 1157–1182.

[20] Xiaofei He, Deng Cai, and Partha Niyogi. 2005. Laplacian Score for Feature Selection. In *NeurIPS*. 507–514.

[21] Wassily Hoeffding. 1963. Probability Inequalities for Sums of Bounded Random Variables. *J. Amer. Statist. Assoc.* 58, 301 (1963), 13–30.

[22] Guanhao Hou, Xingguang Chen, Sibo Wang, and Zhewei Wei. 2021. Massively Parallel Algorithms for Personalized PageRank. *PVLDB* 14, 9 (2021), 1668–1680.

[23] Guanhao Hou, Qintian Guo, Fangyuan Zhang, Sibo Wang, and Zhewei Wei. 2023. Personalized PageRank on Evolving Graphs with an Incremental Index-Update Scheme. *Proc. ACM Manag. Data* 1, 1 (2023), 25:1–25:26.

[24] Jeremy Howard. 2013. The business impact of deep learning. In *SIGKDD*. 1135.

[25] Alicia Key, Bill Howe, Daniel Perry, and Cecilia R. Aragon. 2012. VizDeck: self-organizing dashboards for visual analytics. In *SIGMOD*. 681–684.

[26] Benny Kimelfeld and Yehoshua Sagiv. 2006. Finding and approximating top-k answers in keyword proximity search. In *PODS*. 173–182.

[27] Jundong Li, Kewei Cheng, Suhang Wang, Fred Morstatter, Robert P. Trevino, Jiliang Tang, and Huan Liu. 2018. Feature Selection: A Data Perspective. *ACM Comput. Surv.* 50, 6 (2018), 94:1–94:45.

[28] Xiaolei Li and Jiawei Han. 2007. Mining Approximate Top-K Subspace Anomalies in Multi-Dimensional Time-Series Data. In *PVLDB*. 447–458.

[29] Huan Liu and Rudy Setiono. 1995. Chi2: feature selection and discretization of numeric attributes. In *ICTAI*. 388–391.

[30] Claudio Lucchese, Salvatore Orlando, and Raffaele Perego. 2014. A Unifying Framework for Mining Approximate Top-k Binary Patterns. *IEEE Trans. Knowl. Data Eng.* 26, 12 (2014), 2900–2913.

[31] Frank J Massey Jr. 1951. The Kolmogorov-Smirnov test for goodness of fit. *Journal of the American statistical Association* 46, 253 (1951), 68–78.

[32] Colin McDiarmid et al. 1989. On the method of bounded differences. *Surveys in combinatorics* 141, 1 (1989), 148–188.

[33] Frank Olken. 1993. *Random Sampling from Databases*. Ph. D. Dissertation. University of California at Berkeley.

[34] Frank Olken and Doron Rotem. 1986. Simple Random Sampling from Relational Databases. In *PVLDB*. 160–169.

[35] Yongjoo Park, Barzan Mozafari, Joseph Sorenson, and Junhao Wang. 2018. VerdictDB: Universalizing Approximate Query Processing. In *SIGMOD*. 1461–1476.

[36] Raghu Ramakrishnan, Baskar Sridharan, John R. Douceur, Pavan Kasturi, Balaji Krishnamachari-Sampath, Karthick Krishnamoorthy, Peng Li, Mitica Manu, Spiro Michaylov, Rogério Ramos, Neil Sharman, Zee Xu, Youssef Barakat, Chris Douglas, Richard Draves, Shrikant S. Naidu, Shankar Shastry, Atul Sikaria, Simon Sun, and Ramarathnam Venkatesan. 2017. Azure Data Lake Store: A Hyperscale Distributed File Service for Big Data Analytics. In *SIGMOD*. 51–63.

[37] Mohammad Rastegari, Chen Fang, and Lorenzo Torresani. 2011. Scalable object-class retrieval with approximate and top-k ranking. In *ICCV*. 2659–2666.

[38] Sunita Sarawagi. 2000. User-Adaptive Exploration of Multidimensional Data. In *PVLDB*. 307–316.

[39] Sunita Sarawagi, Rakesh Agrawal, and Nimrod Megiddo. 1998. Discovery-Driven Exploration of OLAP Data Cubes. In *EDBT*, Vol. 1377. 168–182.

[40] Jinwook Seo and Ben Shneiderman. 2005. A rank-by-feature framework for interactive exploration of multidimensional data. *Inf. Vis.* 4, 2 (2005), 96–113.

[41] Konstantin Shvachko, Hairong Kuang, Sanjay Radia, and Robert Chansler. 2010. The Hadoop Distributed File System. In *MSST*. 1–10.

[42] Vishal Sikka, Franz Färber, Wolfgang Lehner, Sang Kyun Cha, Thomas Peh, and Christof Bornhövd. 2012. Efficient transaction processing in SAP HANA database: the end of a column store myth. In *SIGMOD*. 731–742.

[43] Jimeng Sun and Chandan K. Reddy. 2013. Big data analytics for healthcare. In *SIGKDD*. 1525.

[44] Ignacio G. Terrizzano, Peter M. Schwarz, Mary Roth, and John E. Colino. 2015. Data Wrangling: The Challenging Yourney from the Wild to the Lake. In *CIDR*.

[45] Dimitri Theodoratos and Timos K. Sellis. 1997. Data Warehouse Configuration. In *PVLDB*. 126–135.

[46] Hannu Toivonen. 1996. Sampling Large Databases for Association Rules. In *PVLDB*. 134–145.

[47] Manasi Vartak, Sajjadur Rahman, Samuel Madden, Aditya G. Parameswaran, and Neoklis Polyzotis. 2015. SEEDB: Efficient Data-Driven Visualization Recommendations to Support Visual Analytics. *PVLDB* 8, 13 (2015), 2182–2193.

[48] Chi Wang and Kaushik Chakrabarti. 2018. Efficient Attribute Recommendation with Probabilistic Guarantee. In *SIGKDD*. 2387–2396.

[49] Chi Wang and Bailu Ding. 2019. Fast Approximation of Empirical Entropy via Subsampling. In *SIGKDD*. 658–667.

[50] Sibo Wang, Youze Tang, Xiaokui Xiao, Yin Yang, and Zengxiang Li. 2016. HubPPR: Effective Indexing for Approximate Personalized PageRank. *PVLDB* 10, 3 (2016), 205–216.

[51] Sibo Wang and Yufei Tao. 2018. Efficient Algorithms for Finding Approximate Heavy Hitters in Personalized PageRanks. In *SIGMOD*. 1113–1127.

[52] Sibo Wang, Renchi Yang, Runhui Wang, Xiaokui Xiao, Zhewei Wei, Wenqing Lin, Yin Yang, and Nan Tang. 2019. Efficient Algorithms for Approximate Single-Source Personalized PageRank Queries. *ACM Trans. Database Syst.* 44, 4 (2019), 18:1–18:37.

[53] Sibo Wang, Renchi Yang, Xiaokui Xiao, Zhewei Wei, and Yin Yang. 2017. FORA: Simple and Effective Approximate Single-Source Personalized PageRank. In *SIGKDD*. 505–514.

[54] Abdul Wasay, Manos Athanassoulis, and Stratos Idreos. 2015. Queriosity: Automated Data Exploration. In *IEEE Big Data*. 716–719.

[55] Kanit Wongsuphasawat, Dominik Moritz, Anushka Anand, Jock D. Mackinlay, Bill Howe, and Jeffrey Heer. 2016. Voyager: Exploratory Analysis via Faceted Browsing of Visualization Recommendations. *IEEE Trans. Vis. Comput. Graph.* 22, 1 (2016), 649–658.

[56] Lei Yu and Huan Liu. 2003. Feature Selection for High-Dimensional Data: A Fast Correlation-Based Filter Solution. In *ICML*. 856–863.

[57] Chaoqun Zhan, Maomeng Su, Chuangxian Wei, Xiaoqiang Peng, Liang Lin, Sheng Wang, Zhe Chen, Feifei Li, Yue Pan, Fang Zheng, and Chengliang Chai. 2019. AnalyticDB: Real-time OLAP Database System at Alibaba Cloud. *PVLDB* 12, 12 (2019), 2059–2070.