Efficient Approximate Algorithms for Empirical Entropy and Mutual Information

Xingguang Chen
The Chinese University of Hong Kong
China
xgchen@link.cuhk.edu.hk

Sibo Wang
The Chinese University of Hong Kong
China
swang@se.cuhk.edu.hk

ABSTRACT

Empirical entropy is a classic concept in data mining and the foundation of many other important concepts like mutual information. However, computing the exact empirical entropy/mutual information on large datasets can be expensive. Some recent research work explores sampling techniques on the empirical entropy/mutual information to speed up the top-k and filtering queries. However, their solution still aims to return the exact answers to the queries, resulting in high computational costs.

Motivated by this, in this work, we present approximate algorithms for the top-k queries and filtering queries on empirical entropy and empirical mutual information. The approximate algorithm allows user-specified tunable parameters to control the trade-off between the query efficiency and accuracy. We design effective stopping rules to return the approximate answers with improved query time. We further present theoretical analysis and show that our proposed solutions achieve improved time complexity over previous solutions. We experimentally evaluate our proposed algorithms on real datasets with up to 31M records and 179 attributes. Our experimental results show that the proposed algorithm consistently outperforms the state of the art in terms of computational efficiency, by an order of magnitude in most cases, while providing the same accurate result.

CCS CONCEPTS

• **Theory of computation → Design and analysis of algorithms**; **Approximation algorithms analysis**.

KEYWORDS

Empirical Entropy; Empirical Mutual Information; Sampling

ACM Reference Format:
Xingguang Chen and Sibo Wang. 2021. Efficient Approximate Algorithms for Empirical Entropy and Mutual Information. In *Proceedings of the 2021 International Conference on Management of Data (SIGMOD '21), June 20–25, 2021, Virtual Event, China.* ACM, New York, NY, USA, 13 pages. https://doi.org/10.1145/3448016.3457255

SIGMOD '21, June 20–25, 2021, Virtual Event, China

ACM ISBN 978-1-4503-8343-1/21/06…$15.00
https://doi.org/10.1145/3448016.3457255

1 INTRODUCTION

Given a discrete random variable X with a probability mass function $P(X)$, the *entropy* $H(X)$ of the random variable X is defined as $H(X) = \mathbb{E}[-\log_2 P(X)]$. In most real-life applications, the distribution of X is usually unknown, and only the empirical distribution of X can be obtained according to some input data D. The entropy derived according to the empirical distribution of X with input data D is usually defined as the *empirical entropy* of X on D, referred to as $H_D(X)$. In data analysis tasks, the variable X is usually an attribute of the input data, and the empirical distribution of X is then the distribution of each possible attribute value that appears in X. The *empirical mutual information* is a generalized concept to define the empirical entropy when considering multiple attributes of the input data. Both the empirical entropy and mutual information are widely used in real applications. For example, the U.S. Census Bureau provides a public dataset [32] that includes millions of households' records. Each record includes more than 100 attributes like ancestry, education, work, transportation, internet use, residency, and so on. The dataset can be used to build data mining models for many real-life tasks, e.g., to build a classifier to identify high-value insurance consumers. However, due to the curse of dimensionality, the large number of attributes (high dimensionality) usually causes the high training complexity of the prediction models. Feature selection, a core step in data mining, is usually applied to select only the useful and task-relevant attributes to build the models. In the literature, empirical entropy/mutual information [2, 5, 12, 13, 19, 20, 24, 26, 31, 39] is widely used in feature selection. Apart from feature selection, empirical entropy and mutual information further find many applications in IPv6 address analysis [14], decision tree learning [3, 27, 33], graphical model structure learning [10], and categorical clustering [4, 21].

In this paper, we focus on two types of queries: the top-k and filtering queries on empirical entropy and mutual information. For top-k queries, it aims to return the k attributes with the highest empirical entropy/mutual information scores. For the filtering queries, a threshold η is given, and the goal is to return the attributes with empirical entropy/mutual information no smaller than η. A straightforward solution is to derive the exact score by scanning all the records, which is too expensive on large datasets. Luckily, in most real applications, error-bounded approximations of the empirical entropy and mutual information are usually sufficient. For example, in [2, 12, 26, 31] (resp. [19, 24, 39]), the authors show that an approximate solution of the top-k query (resp. filtering query) will be sufficient to do feature selection and provide the useful and task-relevant attributes. With a more efficient approximate solution for top-k and filtering queries, we can significantly reduce the computational costs for the feature selection phase. Therefore, most

existing work focuses on approximate solutions, e.g., [16, 30, 32], to derive error-bounded estimations of the empirical entropy/mutual information. The state-of-the-art solution for top-k and filtering queries is proposed by Wang et al. [32]. In particular, they adaptively sample records, derive tight estimation upper and lower bounds for each attribute, and prune the attributes by the upper and lower bounds. However, they still aim to return attributes that are exactly the top-k or exactly no smaller than the threshold η, making the sampling cost still high and leaving much room for improvement.

Motivated by the deficiency of existing solutions, we propose approximate algorithms for the top-k and filtering queries on empirical entropy and mutual information. For top-k queries, the main deficiency of the state-of-the-art solution in [32] is that they need to sample a sufficiently large number of records to distinguish the lower bound of the attribute with the k-th largest score and the upper bound of the attribute with the $(k+1)$-th largest score. If the gap Δ between the k-th largest score and the $(k+1)$-th largest score is very small, the sampling cost can be rather high. However, in real-life applications, if the gap Δ is very small, it means that the two attributes with the k-th largest and the $(k+1)$-th largest score are similarly important, and returning either one should have almost no impact to the downstream analytic tasks. Therefore, we aim to return an approximate top-k answer, formally defined in Section 2.1, such that the returned k attributes have estimated values that are close to that of the k attributes with the real highest top-k values. For filtering queries, the deficiency of the state-of-the-art solution is that they will strictly return the attributes with scores no smaller than the threshold η. However, in downstream tasks, the attributes close to the threshold should have a negligible effect, since otherwise a more appropriate threshold is expected. Therefore, we aim to answer approximate filtering queries that return attributes with scores larger enough than the threshold and relax the conditions for the attributes with scores close to the threshold.

Since we relax the conditions for top-k and filtering queries, the sampling cost can be significantly reduced as will be shown in our experimental evaluation. The approximate algorithm allows a user-specified error parameter to control the trade-off between the accuracy and efficiency of the returned answer with a strong theoretical guarantee. However, a big challenge is how to design effective stopping conditions to provide query answers that satisfy the approximation guarantee. We tackle this challenging issue for both the top-k and filtering queries and prove that the algorithm returns approximate query answers with high probability. We further show that the time complexity of our proposed algorithms improves over existing alternatives. Extensive experiments on large real datasets show that our proposed algorithms improve over existing solutions by an order of magnitude in most cases without sacrificing the query accuracy.

2 PRELIMINARIES

2.1 Problem Definition

Let $\mathcal{D}$ be a dataset consisting of N records and h attributes. Let $A = \{\alpha_1, \alpha_2, \cdots, \alpha_h\}$ be the set of attributes in $\mathcal{D}$ and $\mathcal{D}(\alpha)$ be the attribute values of all records with respect to attribute α. Given an attribute α, the support size u_α is the number of distinct attribute values appeared in $\mathcal{D}(\alpha)$. We further assume that the attribute values in $\mathcal{D}(\alpha)$ fall into the range of $[1, u_\alpha]$[1], which can be easily handled by a simple one-to-one match preprocessing. Define $n_i(\alpha)$, or simply n_i if the context is clear, as the number of occurrence of attribute value $i \in [1, u_\alpha]$ in $\mathcal{D}(\alpha)$.

Table 1: Frequently used notations.

Notation	Description
$\mathcal{D}$	an input dataset
$\mathcal{S}$	a randomly sampled subset of $\mathcal{D}$
N	the number of records in $\mathcal{D}$
M	the number of records in $\mathcal{S}$
α, A	attribute α from the set A of attributes in $\mathcal{D}$
h	the number of attributes
u_α	the number of distinct values for attribute α
p_f	the probability that the algorithm fails to return an approximate query
$H(\alpha)$	the empirical entropy of α on $\mathcal{D}$
$\underline{H}(\alpha), \overline{H}(\alpha)$	a lower and upper bound of $H_{\mathcal{D}}(\alpha)$
$I(\alpha_1, \alpha_2)$	the empirical mutual information between α_1, α_2 on $\mathcal{D}$
$\underline{I}(\alpha_1, \alpha_2), \overline{I}(\alpha_1, \alpha_2)$	a lower and upper bound of $I_{\mathcal{D}}(\alpha_1, \alpha_2)$
η	the threshold for the filtering query
ϵ	the error parameter for approximate queries

Definition 1 (Empirical entropy). *Given the input dataset $\mathcal{D}$ and an attribute $\alpha \in A$ with support size u_α, the empirical entropy $H_{\mathcal{D}}(\alpha)$ of attribute α with respect to dataset $\mathcal{D}$ is defined as:*

$$H_{\mathcal{D}}(\alpha) = -\sum_{i=1}^{u_\alpha} \frac{n_i}{N} \log_2 \frac{n_i}{N}.$$

Given two input attributes α_1 and α_2, let $n_{i,j}$ be the number of records in $\mathcal{D}$ such that the record has a value of $i \in [1, u_{\alpha_1}]$ on attribute α_1 and a value of $j \in [1, u_{\alpha_2}]$ on attribute α_2. Then, the empirical joint entropy between α_1 and α_2 is defined as:

$$H_{\mathcal{D}}(\alpha_1, \alpha_2) = -\sum_{i=1}^{u_{\alpha_1}} \sum_{j=1}^{u_{\alpha_2}} \frac{n_{i,j}}{N} \log_2 \frac{n_{i,j}}{N}.$$

Definition 2 (Empirical mutual information). *Given the input dataset $\mathcal{D}$ and two attributes α_1, α_2, the empirical mutual information $I_{\mathcal{D}}(\alpha_1, \alpha_2)$ between α_1 and α_2 on $\mathcal{D}$ is defined as:*

$$I_{\mathcal{D}}(\alpha_1, \alpha_2) = H_{\mathcal{D}}(\alpha_1) + H_{\mathcal{D}}(\alpha_2) - H_{\mathcal{D}}(\alpha_1, \alpha_2).$$

Definition 3 (Top-k query). *Given the input data $\mathcal{D}$ and a positive integer k, the top-k query on empirical entropy (resp. empirical mutual information) returns the k attributes with the highest empirical entropy (resp. empirical mutual information) scores.*

Definition 4 (Filtering query). *Given the input data $\mathcal{D}$ and a threshold η, the filtering query on empirical entropy (resp. empirical mutual information) returns all attributes whose empirical entropy (resp. empirical mutual information) score is no smaller than η.*

Note that the empirical mutual information takes two attributes as the input. In real applications, e.g., [12, 26, 31], we are given one

[1]We use $[1, n]$ to indicate the set of integers $\{1, 2, \cdots, n\}$.

attribute α_t and need to select from remaining ones for the top-k or filtering queries. We denote attribute α_t as the target attribute and the set C of remaining attributes as the candidate set.

As we mentioned in Section 1, returning exact answers for top-k or filtering queries still takes a lot of sampling costs. In the meantime, we can relax the conditions of the top-k and filtering query to reduce the sampling costs. Following previous studies on approximate top-k queries [6, 29, 34, 36, 37], the approximate top-k query on empirical entropy/mutual information is defined as follows.

Definition 5 (Approximate top-k query). *Given the input data $\mathcal{D}$, a positive integer k, an error parameter $0 < \epsilon < 1$, and a failure probability p_f, let $s(\alpha)$ be the exact score, either the empirical entropy or mutual information, of attribute α and $\hat{s}(\alpha)$ be the estimated score of $s(\alpha)$. Let $\alpha_1^*, \alpha_2^*, \cdots, \alpha_k^*$ be the k attributes with the top-k scores such that $s(\alpha_1^*) \geq s(\alpha_2^*) \cdots \geq s(\alpha_k^*)$. The approximate top-$k$ query returns k attributes $\alpha_1', \alpha_2', \cdots \alpha_k'$ with the top-k highest estimation scores in order such that the following holds:*

- (i) $\hat{s}(\alpha_i') \geq (1-\epsilon) \cdot s(\alpha_i')$,
- (ii) $s(\alpha_i') \geq (1-\epsilon) \cdot s(\alpha_i^*)$. □

For condition (i), it requires that the estimated scores of the returned k attributes are accurate enough. For condition (ii), it requires that the returned i-th attribute has a score close to the exact i-th largest score. Combining conditions (i) and (ii) assures that the returned approximate top-k answers are close to the exact top-k answers with theoretical guarantees.

For filtering queries, it shares a similar spirit as heavy hitter queries [8, 9, 35] where both require reporting the elements with a score above a threshold value. Following approximate heavy hitters [8, 9], the approximate filtering query is defined as follows.

Definition 6 (Approximate filtering query). *Given the input data $\mathcal{D}$, a threshold η, an error parameter $0 < \epsilon < 1$, and a failure probability p_f, let $s(\alpha)$ be the exact score, either the empirical entropy or mutual information, of attribute α. The approximate filtering query returns a set $X \subseteq A$ such that:*

- *if $s(\alpha) \geq (1+\epsilon)\eta$, α must belong to X;*
- *else if $(1+\epsilon)\eta > s(\alpha) \geq (1-\epsilon)\eta$, α may or may not belong to X;*
- *otherwise, α should not belong to X.*

The above definitions allow users to control the quality of the top-k and filtering query answer by the error parameter ϵ. When ϵ is very small, then the returned attributes have similarly high quality results as the real top-k or filtering answers. As we will see in our theoretical analysis, when ϵ becomes smaller, it incurs higher sampling costs. There is a trade-off between the query efficiency and result accuracy. We will evaluate the impact of ϵ in Section 6.

Remark. Notice that in the following sections, we will omit the subscript $\mathcal{D}$ and simply use $H(\alpha)$ or $I(\alpha_1, \alpha_2)$ to denote the empirical entropy and empirical mutual information on the input dataset $\mathcal{D}$ if the context is clear. Besides, we will only require our algorithm to return an approximate query answer with a probability of $1 - p_f$.

Table 1 lists the frequently used notations in the paper.

2.2 Existing Solutions

A straightforward solution is an exact method that scans all the records column by column (assuming that the data is stored in column style). With the exact scores, the top-k or filtering query answer can be returned. However, such a method is expensive when the datasets become huge and include many attributes. This motivates the state-of-the-art sampling-based solution in [32].

EntropyRank and EntropyFilter. The main idea of EntropyRank and EntropyFilter proposed by Wang et al. [32] is to sample a subset $\mathcal{S}$ of the records from the input dataset $\mathcal{D}$ and then derive an estimation of the empirical entropy or mutual information. Given an attribute α, a sampled subset $\mathcal{S} \subseteq \mathcal{D}$, let $\mathcal{S}(\alpha)$ be the attribute values of all sampled records in $\mathcal{S}$ with respect to attribute α. The empirical entropy of $\mathcal{S}(\alpha)$ on attribute α with respect to $\mathcal{S}$ can be similarly defined. In particular, let M be the number of records in the sampled subset $\mathcal{S}$. Let $m_i(\alpha)$, or simply m_i if the context is clear, be the number of occurrence of attribute value $i \in [1, u_\alpha]$ in $\mathcal{S}(\alpha)$. The empirical entropy $H_{\mathcal{S}}(\alpha)$ is defined as:

$$H_{\mathcal{S}}(\alpha) = -\sum_{i=1}^{u_\alpha} \frac{m_i}{M} \log_2 \frac{m_i}{M}. \tag{1}$$

The empirical mutual information $I_{\mathcal{S}}(\alpha_1, \alpha_2)$ can be similarly defined. However, $H_{\mathcal{S}}(\alpha)$ (resp. $I_{\mathcal{S}}(\alpha_1, \alpha_2)$) is not an unbiased estimation of $H_{\mathcal{D}}(\alpha)$ (resp. $I_{\mathcal{D}}(\alpha_1, \alpha_2)$) and there exists some gap between $H_{\mathcal{D}}(\alpha)$ (resp. $I_{\mathcal{D}}(\alpha_1, \alpha_2)$) and the expectation of $H_{\mathcal{S}}(\alpha)$ (resp. $I_{\mathcal{S}}(\alpha_1, \alpha)$) over a random choice of subset $\mathcal{S}$ of size M. As proved in [32], such a gap can be bounded by the following lemma.

Lemma 1. *Let $\mathcal{S}$ be a subset of size M randomly sampled from $\mathcal{D}$. Let u_α be the support size of attribute α. The following holds:*

$$0 \leq H_{\mathcal{D}}(\alpha) - \mathbb{E}[H_{\mathcal{S}}(\alpha)] \leq \log_2 \left(1 + \frac{(u_\alpha - 1)(N - M)}{M(N-1)}\right). \tag{2}$$

Therefore, as long as the gap between $H_{\mathcal{S}}(\alpha)$ and $\mathbb{E}[H_{\mathcal{S}}(\alpha)]$ can be bounded, an error-bounded estimation of $H_{\mathcal{D}}(\alpha)$ can be derived. Notice that the classic concentration bounds like Chernoff bound [23] or McDiarmid's inequality [22] cannot be applied to bound the gap between the estimation and its expectation for $H_{\mathcal{S}}(\alpha)$. To explain, the empirical entropy cannot be expressed as a mean of samples therefore Chernoff bound cannot be applied. Furthermore, the sample subset $\mathcal{S}$ is a sample without replacement from $\mathcal{D}$ while the McDiarmid's inequality considers sampling with replacement from a distribution. To tackle this issue, Wang et al. [32] explores the concentration bounds for sampling without replacement. In particular, the randomly sampled subset $\mathcal{S}$ of size M can be regarded as the first M records after a random shuffle to the input data $\mathcal{D}$. Let $\pi(\mathbf{Z}) \triangleq (Z_1, Z_2, \cdots, Z_n)$ be the permutation vector over the input $\mathcal{D}$ after the random shuffle where $Z_i \in [1, N]$ is the index of the i-th element in the shuffled data. A function $f\colon \pi(\mathbf{Z}) \to \mathbb{R}$, is called (M, N)-*symmetric* with respect to the permutation $\pi(\mathbf{Z})$ if f does not change its value under the change of the order of the first M elements or the last $N - M$ elements.

Obviously, the empirical entropy and the empirical mutual information over $\mathcal{D}$ is (M, N)-symmetric. Notice that, given a randomly sampled subset $\mathcal{S}$ of size M corresponding the first M elements in the permutation, $H_{\mathcal{S}}(\alpha)$ and $I_{\mathcal{S}}(\alpha_1, \alpha_2)$ are also (M, N)-symmetric.

Given a (M, N)-symmetric function f, we have the following concentration bound for $f(\mathbf{Z})$ and its expectation $\mathbb{E}[f(\mathbf{Z})]$. Let $\mathbf{Z}^{i,j}$ be a perturbed permutation vector obtained by exchanging only

Z_i and Z_j in $\mathbf{Z}$. The following (rephrased) concentration bound is proposed by El-Yaniv and Pechyony [11].

Lemma 2 ([11]). *Let $\mathbf{Z}$ be a random permutation over input $\mathcal{D}$ and $f(\mathbf{Z})$ be a (M, N)-symmetric function with $|f(\mathbf{Z}) - f(\mathbf{Z}^{i,j})| < \beta$ for all $i \in [1, M]$ and $j \in [M+1, N]$. Then, for any $\lambda > 0$, we have that:*

$$\Pr[f(\mathbf{Z}) - \mathbb{E}[f(\mathbf{Z})] \geq \lambda] \leq \exp\left(-\frac{2\lambda^2}{M\beta^2}\left(\frac{N-1/2}{N-M}\right)\left(1 - \frac{1}{2\max(M, N-M)}\right)\right). \quad (3)$$

According to [32], $|f(\mathbf{Z}) - f(\mathbf{Z}^{i,j})| < 2\log_2 M/M$. Let $f(\mathbf{Z})$ be $H_{\mathcal{S}}(\alpha)$ in Equation 1 where the empirical entropy only considers the first M records and discards records from $M+1$ to N. Given Equations 2-3, one can derive an upper bound of $H_{\mathcal{D}}(\alpha)$. To derive a lower bound, we can set $f(\mathbf{Z}) = -H_{\mathcal{S}}(\alpha)$ and then combine Equations 2-3. Notice that with an increased size M of the sampled subset $\mathcal{S}$, the tighter lower and upper bounds we have for $H_{\mathcal{D}}(\alpha)$. When $M = N$, then we derive the exact value of $H_{\mathcal{D}}(\alpha)$.

To answer the top-k query, EntropyRank runs in batches. In the i-th iteration, it samples without replacement a batch of size b subset from $\mathcal{D}$ and combines with the previously sampled $(i-1)\cdot b$ records. Then, with the $i \cdot b$ records, it derives the lower and upper bound for each attribute α and examines if the k-th largest lower bound is no smaller than the $(k+1)$-th largest upper bound. If the answer is yes, it returns the k nodes with the largest k lower bound. Otherwise, it turns to another iteration to derive tighter upper and lower bounds for each attribute. Similarly, for the filtering queries, it runs in batches until we can identify if the upper bound of $H_{\mathcal{D}}(\alpha)$ is smaller than η or the lower bound of $H_{\mathcal{D}}(\alpha)$ is larger than η.

As we mentioned, the main deficiency of EntropyRank and EntropyFilter is that they always return the exact answers for the top-k queries and the exact filtering queries. In reality, an approximate answer is sufficient, which motivates us to design the algorithms for approximate top-k and filtering queries.

3 PROPOSED SOLUTION

In the literature, there exists a plethora of research work focusing on the approximate top-k query processing, e.g., [6, 29, 36], or heavy hitter queries, e.g., [8, 9], that share similar spirits as the filtering queries by reporting the elements above a certain threshold. However, to the best of our knowledge, all these studies consider only the sum or average of random variables. While in our problem setting, the empirical entropy cannot be expressed as either the sum or the average of random variables, making it more challenging than existing problems. Motivated by this, we propose our framework *SWOPE*[2] to answer the approximate queries efficiently. For the ease of exposition, we focus on empirical entropy in this section. We present our approximate top-k and filtering query algorithm on empirical entropy in Section 3.1 and Section 3.2, respectively. We will show how to extend the algorithms to empirical mutual information in Section 4.

3.1 Approximate Top-k Query Processing

Recap that in the top-k query on the empirical entropy, we are given an input dataset $\mathcal{D}$, and the goal is to find k attributes from A that have the top-k empirical entropy. Our main idea is to increase the sample size adaptively, thus deriving tighter and tighter bound, and check if the conditions are met to return the approximate top-k answer satisfying Definition 5. If the conditions are not met, the size of the sampled subset is doubled and the algorithm terminates until the top-k answer satisfies the approximation guarantee.

[2] Sampling Without Replacement for Empirical Entropy.

Algorithm 1: SWOPE-Top-k: Empirical Entropy

Input: Dataset $\mathcal{D}$, k, p_f, ϵ
Output: An approximate top-k query answer

```
C ← A, M ← M_0, R ← ∅, i_max ← ⌈log_2 N/M_0⌉ + 1,
  p'_f ← p_f / (i_max · h);
while M ≤ N do
    for α ∈ C do
        Calculate H_(α), H‾(α), b(α) and λ by Lemma 3 with
          p ← p'_f;
    R ← top-k attributes from C according to H‾(α);
    H‾(α'_k) ← the k-th largest H‾(α) for α ∈ C;
    b_max ← the largest b(α) for α ∈ R;
    if (H‾(α'_k) − 2λ − b_max)/H‾(α'_k) ≥ 1 − ε then
        return R;
    else if M < N then
        M ← min{N, 2M};
    else
        break;
    H_(α''_k) ← the k-th largest H_(α) for α ∈ C;
    for α ∈ C do
        if H‾(α) < H_(α''_k) then
            C ← C \ {α};
return R;
```

To examine the conditions, with a sampled subset $\mathcal{S}$, for each attribute α, we estimate $H(\alpha)$ by calculating the lower bound $\underline{H}(\alpha)$ and upper bound $\overline{H}(\alpha)$ with the concentration bound in Lemma 2 so that $H(\alpha) \in [\underline{H}(\alpha), \overline{H}(\alpha)]$ with high probability. Notice that with a larger size of the sampled subset $\mathcal{S}$, the upper and lower bounds become tighter, and the estimated result is more accurate. With the estimated results, we devise stopping conditions to guarantee that when the algorithm terminates, it returns an approximate top-k answer satisfying conditions in Definition 5 with high probability.

One of the main challenges is that in approximate top-k answers, the exact empirical entropy of the returned attribute α'_i with the i-th largest estimation, i.e., $H(\alpha'_i)$, should be no smaller than $(1-\epsilon)\cdot H(\alpha^*_i)$ where α^*_i is the attribute with the exact i-th largest score which is unknown in advance. Therefore, how to design an effective stopping condition is very important. If it stops too early, the approximation ratio may not be satisfied; if it stops too late, it incurs additional running time, sacrificing the performance. Another challenge is how to bound the expected running time. In [32], the expected running time of the proposed top-k algorithm is linear to $\frac{1}{\Delta^2}$ where Δ is the gap between the k-th and $(k+1)$-th largest empirical entropy. When Δ is very small, the algorithm may incur

a high sampling cost. Can we achieve a better expected running time? Finally, there exist dependencies between the samples among different iterations. Does the concentration bound still hold in such cases? Next, we will elaborate on our solution and show how to tackle such challenging issues.

Deriving lower and upper bounds. To provide bounds for the empirical entropy $H(\alpha)$, i.e., $[\underline{H}(\alpha), \overline{H}(\alpha)]$, we can derive a connection between the upper (resp. lower) bound $\overline{H}(\alpha)$ (resp. $\underline{H}(\alpha)$) and the size M of the sampled set $\mathcal{S}$ by exploring Lemma 2. Let $\beta = \log_2 \frac{M}{M-1} + \frac{\log_2(M-1)}{M}$. We have the following lemma for the upper and lower bounds.

Lemma 3. *Given attribute α with support size u_α, a random subset $\mathcal{S}$ of size M, and a failure probability p, we have that:*

$$H(\alpha) \geq \underline{H}(\alpha) \triangleq H_{\mathcal{S}}(\alpha) - \beta \sqrt{\frac{M(N-M)\ln(2/p)}{2(N-1/2)\left(1-\frac{1}{2\max(M,N-M)}\right)}} \tag{4}$$

$$H(\alpha) \leq \overline{H}(\alpha) \triangleq H_{\mathcal{S}}(\alpha) + \beta \sqrt{\frac{M(N-M)\ln(2/p)}{2(N-1/2)\left(1-\frac{1}{2\max(M,N-M)}\right)}} + \log_2\left(1+\frac{(u_\alpha-1)(N-M)}{M(N-1)}\right). \tag{5}$$

both hold with probability at least $1-p$.

We further define λ and b as follows:

$$\lambda = \beta \sqrt{\frac{M(N-M)\ln(2/p)}{2(N-1/2)\left(1-\frac{1}{2\max(M,N-M)}\right)}}, \tag{6}$$

$$b(\alpha) = \log_2\left(1+\frac{(u_\alpha-1)(N-M)}{M(N-1)}\right), \tag{7}$$

which will be frequently used later. With the upper and lower bounds of $H(\alpha)$, we are ready to introduce our algorithm for approximate entropy top-k query.

Main algorithm. Algorithm 1 shows the pseudo-code of our approximate top-k algorithm for empirical entropy. At the beginning, we initialize a candidate set C to include all the attributes in A. Then, the algorithm runs in iterations. In the first iteration, it samples M_0 records. The setting of M_0 will be discussed later. Then, in each iteration, it calculates the lower bound $\underline{H}(\alpha)$, upper bound $\overline{H}(\alpha)$, λ, and $b(\alpha)$ for each attribute in C (Algorithm 1 Lines 3-4). Next, it retrieves a set R of the k attributes with the top-k largest upper bounds $\overline{H}(\alpha)$ among all $\alpha \in C$ (Algorithm 1 Line 6). Denote α'_k as the attribute with the k-th largest upper bound and $b_{\max} = \max_{\alpha \in R} b(\alpha)$ (Algorithm 1 Lines 6-7). Then, the algorithm checks if R is an approximate top-k answer or not. The stopping condition is quite simple: if $\overline{H}(\alpha'_k) - 2\lambda - b_{\max})/\overline{H}(\alpha'_k) \geq 1-\epsilon$, then the algorithm finishes and returns R (Algorithm 1 Lines 8-9). The correctness of this termination condition will be proved shortly. If the stopping condition is not met and the sample size M is smaller than N, then M is doubled for the next iteration (Algorithm 1 Lines 10-11). Otherwise, if $M = N$, we have already derived the exact answers of the empirical entropy for all attributes in C. It then simply return the k attributes with the highest estimation scores (Algorithm 1 Line 18). We further prune the attributes in C whose upper bound $\overline{H}(\alpha)$ is smaller than $\underline{H}(\alpha''_k)$ (Algorithm 1 Lines 15-17), which cannot be the top-k answers.

Theoretical analysis. It remains to clarify whether Algorithm 1 returns the answer satisfying the definition of the approximate top-k query. The following lemma shows that our algorithm returns an approximate top-k answer with high probability.

Theorem 1. *Let $R = \{\alpha'_1, \alpha'_2, \cdots, \alpha'_k\}$ be k attributes returned by Algorithm 1 sorted in descending order of their upper bounds. Then R is an approximate top-k answer satisfying Definition 5 with at least $1-p_f$ probability.*

Proof. Let α^*_i be the attribute with the exact i-th largest empirical entropy. In each iteration, since the sample size M is fixed, all $\alpha \in C$ use the same λ according to the definition in Equation 6. Equations 4-5 show that for the candidate attribute α, we have

$$\underline{H}(\alpha) = \overline{H}(\alpha) - 2\lambda - b(\alpha).$$

The algorithm terminates when $(\overline{H}(\alpha'_k) - 2\lambda - b_{\max})/\overline{H}(\alpha'_k) \geq 1-\epsilon$. Note that $(\overline{H}(\alpha'_i) - 2\lambda - b_{\max})/\overline{H}(\alpha'_i)$ is monotonic increasing with $\overline{H}(\alpha'_i)$ and therefore $(\overline{H}(\alpha'_i) - 2\lambda - b_{\max})/\overline{H}(\alpha'_i) \geq 1-\epsilon$ for all $\alpha'_i \in R$. Define the estimation of $H(\alpha'_i)$ as $\hat{H}(\alpha'_i) = (\underline{H}(\alpha'_i) + \overline{H}(\alpha'_i))/2 \geq \underline{H}(\alpha'_i)$. Since $\hat{H}(\alpha'_i) \geq \underline{H}(\alpha'_i)$, we have that:

$$\begin{aligned}\hat{H}(\alpha'_i) &\geq \underline{H}(\alpha'_i) = \overline{H}(\alpha'_i) - 2\lambda - b(\alpha'_i) \\ &\geq \overline{H}(\alpha'_i) - 2\lambda - b_{\max} \geq (1-\epsilon)\overline{H}(\alpha'_i) \geq (1-\epsilon)H(\alpha'_i),\end{aligned}$$

where $H(\alpha'_i)$ is the exact empirical entropy score of attribute α'_i. Therefore, the returned k attributes satisfy the first condition of approximate top-k query in Definition 5.

We then show that the returned attributes will satisfy the second condition in Definition 5. From the above analysis, we have that:

$$\underline{H}(\alpha'_i) \geq (1-\epsilon)\overline{H}(\alpha'_i) \text{ for } i = 1, 2, \cdots, k.$$

It is clear that $\underline{H}(\alpha'_i) \leq H(\alpha'_i) \leq \overline{H}(\alpha'_i)$ and $H(\alpha^*_i) \leq \overline{H}(\alpha'_i)$ since $\overline{H}(\alpha'_i)$ is the i-th largest upper bound. Then we have that:

$$H(\alpha'_i) \geq \underline{H}(\alpha'_i) \geq (1-\epsilon)\overline{H}(\alpha'_i) \geq (1-\epsilon)H(\alpha^*_i)$$

satisfying the second requirement of Definition 5.

The above analysis assumes that the derived upper and lower bounds hold for each attribute in each iteration. Notice that the probability of $H(\alpha_t, \alpha) \notin [\underline{H}(\alpha_t, \alpha), \overline{H}(\alpha_t, \alpha)]$ is at most p'_f according to Algorithm 1 Line 4. Since there are at most $i_{\max} = \lceil \log(N/M_0) \rceil + 1$ iterations and there are at most h candidate attributes in total, the total fail probability is at most $i_{\max} \cdot h \cdot p'_f = p_f$ by union bound. We conclude that Algorithm 1 returns an answer satisfying Definition 5 with at least $1-p_f$ probability. □

Another advantage of our proposed algorithm is that the expected running time adaptively depends on the k-th largest empirical entropy score (even though it is unknown). In particular, we have the following theorem on the time complexity of Algorithm 1.

Theorem 2. *The expected running time of Algorithm 1 is:*

$$O\left(\min\left\{hN, \frac{h\log\left(h\log N/p_f\right)\log^2 N}{\epsilon^2 H^2\left(\alpha^*_k\right)}\right\}\right).$$

The proof of Theorem 2 is deferred to Section 5. The above theorem essentially states that the larger the k-th largest empirical entropy it is, the more efficient our top-k algorithm is. If the k-th largest score is a constant, then our time complexity only depends on $O\left(h\log(h\log N/p_f)\log^2 N/\epsilon^2\right)$, which significantly improves over the exact solution $O(h\cdot N)$. Compared to EntropyRank [32], our solution linearly depends on $\frac{1}{H^2(\alpha_k^*)}$ while EntropyRank has a time complexity of $O\left(\frac{h\log(h\cdot N)\log^2 N}{\Delta^2}\right)$, where Δ is the difference between the k-th largest and the $(k+1)$-th largest empirical entropy, and is strictly smaller than $H(\alpha_k^*)$. Therefore, our time complexity is asymptotically better that of EntropyRank.

Theorem 2 further provides a lower bound on the sample size required, that helps us determine the initial sample size M_0. Let $u_{\max}$ be the maximum support size among all attributes in A. Then, the k-th largest empirical entropy can be bounded by $\log_2 u_{\max}$ for any choice of k. Therefore, we set M_0 as:

$$M_0 = \frac{\log\left(h\log N/p_f\right)\log^2 N}{\left(\log_2 u_{\max}\right)^2},$$

which is the minimum number of samples required when the k-th largest empirical entropy is the largest possible value $\log_2 u_{\max}$ and ϵ is the largest value 1.

Dependencies among different iterations. Another issue that is easy to be neglected is the dependency among different samples. According to our algorithm, we will make use of the first $M_0\cdot 2^{i-1}$ records in the i-th iteration. However, the concentration bound in Lemma 2 requires that the subset $\mathcal{S}$ is randomly sampled from $\mathcal{D}$ while the records sampled in the i-th iteration depend on the records sampled in the first $(i-1)$ iterations.

We note that Lemma 2 actually requires a more relaxed condition than randomly sampling a subset from $\mathcal{D}$ and allows more dependencies. In particular, given a sequence of sampled records without replacement $Z=(X_1,X_2,X_3,\cdots,X_N)$, it suffices if $W_0=\mathbb{E}[f(Z)], W_1=\mathbb{E}[f(Z)|X_1],\cdots,W_i=\mathbb{E}[f(Z)|X_1,X_2,\cdots,X_i],\cdots W_n=\mathbb{E}[f(Z)|X_1,X_2,\cdots,X_n]$ forms a martingale process, i.e., $\mathbb{E}[W_{i+1}|X_1,X_2,\cdots,X_i]=W_i$.

To prove $\mathbb{E}[W_{i+1}|X_1,X_2,\cdots,X_i]=W_i$, we use the law of total expectation, i.e., $\mathbb{E}[V|Y]=\mathbb{E}[\mathbb{E}[V|U,Y]|Y]$. In particular, we have:

$\mathbb{E}[W_{i+1}|X_1,X_2,\cdots,X_i]=$

$\mathbb{E}[\mathbb{E}[f(Z)|X_1,X_2,\cdots,X_{i+1}]|X_1,X_2,\cdots,X_i]$ (Definition of W_{i+1})

$=\mathbb{E}[f(Z)|X_1,X_2,\cdots,X_i]$ ($V=f(Z), Y=X_1,X_2,\cdots,X_i, U=X_{i+1}$)

$=W_i$ (Definition of W_i)

Therefore, $W_0,W_1,\cdots,W_N$ forms a martingale process, and the concentration bound in Lemma 2 can still be applied even though there exist dependencies among the samples in different iterations.

3.2 Approximate Filtering Query Processing

Recap that in the filtering query, we are given a dataset $\mathcal{D}$, a threshold η, and the goal is to find attributes in A whose empirical entropy are no less than η. Similar to our top-k processing, we still adaptively increase the sample size step by step until the stopping condition is satisfied. The main challenge is still how to design effective stopping conditions. If we include an attribute α as one of the results only when $\underline{H}(\alpha)>\eta$ or discard it only when $\overline{H}(\alpha)<\eta$, then the algorithm will return the exact answer. This is how the stopping condition is designed in EntropyFilter [32]. The expected running time of EntropyFilter is linear to $\frac{1}{\delta^2}$, where δ is the gap between the score and the threshold η. Intuitively, the smaller η is, the more difficult it is to distinguish the values of η and $H(\alpha)$, which will lead to higher sampling cost. Next, we will introduce our solution, which relaxes the conditions for the attributes close to the threshold, thus significantly increasing the query performance.

Algorithm 2: SWOPE-Filtering: Empirical Entropy

Input: Dataset $\mathcal{D}$, η, p_f, ϵ
Output: An approximate filtering query answer

1 $C\leftarrow A$, $M\leftarrow M_0$, $R\leftarrow\emptyset$, $i_{\max}\leftarrow\lceil\log_2\frac{N}{M_0}\rceil+1$, $p_f'\leftarrow\frac{p_f}{i_{\max}h}$;
2 **while** $C\neq\emptyset$ **and** $M\leq N$ **do**
3 **for** $\alpha\in C$ **do**
4 Calculate $\underline{H}(\alpha),\overline{H}(\alpha)$ by Lemma 3 with $p\leftarrow p_f'$;
5 $\hat{H}(\alpha)\leftarrow(\underline{H}(\alpha)+\overline{H}(\alpha))/2$;
6 **if** $\overline{H}(\alpha)-\underline{H}(\alpha)<2\epsilon\eta$ **then**
7 **if** $\hat{H}(\alpha)\geq\eta$ **then**
8 $R\leftarrow R\cup\{\alpha\}$;
9 $C\leftarrow C\setminus\{\alpha\}$;
10 **else if** $\underline{H}(\alpha)\geq(1-\epsilon)\eta$ **then**
11 $R\leftarrow R\cup\{\alpha\}$;
12 $C\leftarrow C\setminus\{\alpha\}$;
13 **else if** $\overline{H}(\alpha)<(1+\epsilon)\eta$ **then**
14 $C\leftarrow C\setminus\{\alpha\}$;
15 $M\leftarrow\min\{N,2M\}$;
16 **return** R;

Main algorithm. Algorithm 2 shows the pseudo-code of our approximate filtering algorithm on empirical entropy. We first initialize a candidate set C to include all attributes in A and set the sample size $M=M_0$ (Algorithm 2 Line 1). Next, the algorithm runs in iterations and in each iteration, it goes through the attribute values for each $\alpha\in C$ one by one. For each attribute α, the lower bound $\underline{H}(\alpha)$ and upper bound $\overline{H}(\alpha)$ are calculated based on the M samples (Algorithm 2 Line 4). The average of $\underline{H}(\alpha)$ and $\overline{H}(\alpha)$ is defined as $\hat{H}(\alpha)$, which is the estimation of $H(\alpha)$.

Next, the algorithm determines if the attribute α belongs to the answer set or not (Algorithm 2 Lines 6-14). Firstly, it checks if the difference between $\overline{H}(\alpha)$ and $\underline{H}(\alpha)$ is strictly smaller than $2\epsilon\eta$, in which case the bound is tight enough, and we will prune it from C; we add α to the result set R only if $\hat{H}(\alpha)$ is larger than the threshold η. Next, it checks if $\underline{H}(\alpha)$, the lower bound of $H(\alpha)$ is larger than $(1-\epsilon)\eta$. If so, we add it to the result set R and prune it from C. Furthermore, if $\overline{H}(\alpha)$ is strictly smaller than $(1+\epsilon)\eta$, it is pruned from C since it is too small. If the candidate set is still not empty, the sample size is doubled for the next iteration (Algorithm 2 Line 15). The iterations repeat until C becomes empty or N records

have been sampled. Finally, we return the set R of attributes as the approximate filtering query answer.

Theoretical analysis. Next, we show that Algorithm 2 returns an approximate filtering query answer by the following theorem.

Theorem 3. *Let R be the set of attributes returned by Algorithm 2. Then R is an approximate filtering answer satisfying Definition 6 with at least $1 - p_f$ probability.*

Proof. Let α be an arbitrary attribute in C. Without loss of generality, we omit the analysis when $(1-\epsilon)\eta \le H(\alpha) < (1+\epsilon)\eta$, since whether we return it or not the result will meet the conditions in Definition 6. We discuss three possible cases that α will be pruned from the candidate set C.

- **Case 1: $\overline{H}(\alpha) - \underline{H}(\alpha) < 2\epsilon\eta$.** If $H(\alpha) \ge (1+\epsilon)\eta$, then $\overline{H}(\alpha) \ge (1+\epsilon)\eta$. The estimation $\hat{H}(\alpha)$ satisfies that:

$$\hat{H}(\alpha) = \overline{H}(\alpha) - \frac{1}{2}\left(\overline{H}(\alpha) - \underline{H}(\alpha)\right) \ge (1+\epsilon)\eta - \epsilon\eta = \eta.$$

  So α will be included in R, satisfying conditions in Definition 6. If $H(\alpha) < (1-\epsilon)\eta$, then $\underline{H}(\alpha) < (1-\epsilon)\eta$. It holds that:

$$\hat{H}(\alpha) = \underline{H}(\alpha) + \frac{1}{2}\left(\overline{H}(\alpha) - \underline{H}(\alpha)\right) < (1-\epsilon)\eta + \epsilon\eta = \eta.$$

  So α will not be returned by Algorithm 2, satisfying Definition 6.
- **Case 2: $\underline{H}(\alpha) \ge (1-\epsilon)\eta$.** In this case, it will be included in R since $H(\alpha) \ge (1-\epsilon)\eta$, and including α to R will not violate the requirement in Definition 6.
- **Case 3: $\overline{H}(\alpha) < (1+\epsilon)\eta$.** In this case, it will not be included in R since $H(\alpha) < (1+\epsilon)\eta$, and ignoring α will not violate the requirement in Definition 6.

Hence, the removal of α from candidate set C under any of the three conditions will not violate Definition 6.

The above analysis assumes that all upper and lower bounds of $H(\alpha)$ for $\alpha \in C$ are correct. Similar to the analysis of our top-k algorithm, there are total $i_{\max} = \lceil \log(N/M_0) \rceil + 1$ iterations, and we apply the bounds for at most h attributes in each iteration. For each attribute, the bounds fail with at most p'_f probability based on the setting of p in Algorithm 2 Line 4. The total failure probability is at most $i_{\max} h \cdot p'_f = p_f$ by union bound. This finishes the proof. □

We have the following theorem to bound the expected running time of our approximate filtering query on empirical entropy.

Theorem 4. *The expected running time of Algorithm 2 is*

$$O\left(\min\left\{hN, \frac{h\log\left(h\log N/p_f\right)\log^2 N}{\epsilon^2\eta^2}\right\}\right).$$

The proof of Theorem 4 is deferred to Section 5. The time complexity of EntropyFilter [32] is $O\left(\frac{h\log(h\cdot N)\log^2 N}{\delta^2}\right)$, where δ is the smallest gap between the empirical score and η. Then, obviously, δ is strictly smaller than that of η. Hence our algorithm achieves an asymptotic better time complexity than EntropyFilter. Our choice of M_0 is the same as that of top-k by setting $\eta = \log_2 u_{\max}$ and $\epsilon = 1$, where $u_{\max}$ is the maximum support size among all attributes.

4 EXTENSION TO EMPIRICAL MUTUAL INFO.

4.1 Approximate Top-k Query Processing

In the top-k query on empirical mutual information, we have additional input, the target attribute α_t, compared to that on empirical entropy. Recap that the empirical mutual information $I(\alpha_t, \alpha)$ is:

$$I(\alpha_t, \alpha) = H(\alpha_t) + H(\alpha) - H(\alpha_t, \alpha).$$

Therefore, to derive the upper and lower bound of $I(\alpha_t, \alpha)$, we need to derive the upper and lower bounds for $H(\alpha_t)$, $H(\alpha)$, and $H(\alpha_t, \alpha)$. It is not difficult to apply Lemma 3 to derive the upper and lower bounds for $H(\alpha_t)$ and $H(\alpha)$.

For the joint empirical entropy $H(\alpha_t, \alpha)$, to derive lower and upper bounds of $H(\alpha_t, \alpha)$, we bound the gap between $H(\alpha_t, \alpha)$ and its expectation $\mathbb{E}[H_{\mathcal{S}}(\alpha_t, \alpha)]$. According to Lemma 1, we have that:

$$0 \le H_{\mathcal{D}}(\alpha_t, \alpha) - \mathbb{E}[H_{\mathcal{S}}(\alpha_t, \alpha)] \le \log_2\left(1 + \frac{(u_{\alpha_t,\alpha} - 1)\cdot(N-M)}{M\cdot(N-1)}\right),$$

where $u_{\alpha_t,\alpha}$ is the number of distinct pairs between α_t and α in $\mathcal{D}$. Since it is impractical to record the exact value of $u_{\alpha_t,\alpha}$ for all possible combinations of different pairs of attributes in advance, we use an upper bound $\overline{u}_{\alpha_t,\alpha}$ of $u_{\alpha_t,\alpha}$ where

$$\overline{u}_{\alpha_t,\alpha} = u_{\alpha_t} \cdot u_{\alpha},$$

by considering the worst case that all combinations between α_t and α appear in $\mathcal{D}$. So we have that:

$$0 \le H(\alpha_t, \alpha) - \mathbb{E}[H_{\mathcal{S}}(\alpha_t, \alpha)] \le \log_2\left(1 + \frac{(\overline{u}_{\alpha_t,\alpha} - 1)(N-M)}{M(N-1)}\right).$$

Similar to the definition of $b(\alpha)$, we define $b(\alpha_t, \alpha)$ as:

$$b(\alpha_t, \alpha) = \log_2\left(1 + \frac{(\overline{u}_{\alpha_t,\alpha} - 1)(N-M)}{M(N-1)}\right)$$

Then, we have that:

$$H(\alpha_t, \alpha) \ge \underline{H}(\alpha_t, \alpha) \triangleq H_{\mathcal{S}}(\alpha_t, \alpha) - \lambda$$
$$H(\alpha_t, \alpha) \le \overline{H}(\alpha_t, \alpha) \triangleq H_{\mathcal{S}}(\alpha_t, \alpha) + \lambda + b(\alpha_t, \alpha)$$

With the lower and upper bounds of $H(\alpha_t)$, $H(\alpha)$ and $H(\alpha_t, \alpha)$, we can define the lower and upper bounds of $I(\alpha_t, \alpha)$ as:

$$\underline{I}(\alpha_t, \alpha) = \underline{H}(\alpha_t) + \underline{H}(\alpha) - \overline{H}(\alpha_t, \alpha),$$
$$\overline{I}(\alpha_t, \alpha) = \overline{H}(\alpha_t) + \overline{H}(\alpha) - \underline{H}(\alpha_t, \alpha).$$

If the bounds for $H(\alpha_t)$, $H(\alpha)$ and $H(\alpha_t, \alpha)$ each hold with $1-p$ probability, then we have that $I(\alpha_t, \alpha) \in [\underline{I}(\alpha_t, \alpha), \overline{I}(\alpha_t, \alpha)]$ holds with $1 - 3\cdot p$ probability by union bound. We also define $\hat{I}(\alpha_t, \alpha) = (\underline{I}(\alpha_t, \alpha) + \overline{I}(\alpha_t, \alpha))/2$ as the estimated value of $I(\alpha_t, \alpha)$.

Main algorithm. The pseudo-code of the top-k algorithm is shown in Algorithm 3. It shares a similar spirit as the top-k algorithm for empirical entropy. Initially, the candidate set C includes all attributes except the target attribute α_t and samples M_0 records. The setting of M_0 is the same as that Algorithm 1. Then, in each iteration, it derives upper and lower bounds of $I(\alpha_t, \alpha)$ for each attribute in C (Algorithm 3 Lines 3-6) and obtains the k attributes with the highest upper bounds in C and set it to R. Next, it examines if the stopping condition is satisfied. In particular, it checks if

$$(\overline{I}(\alpha_t, \alpha'_k) - 6\lambda - b'_{\max})/\overline{I}(\alpha_t, \alpha'_k) \ge 1 - \epsilon,$$

Algorithm 3: SWOPE-Top-k: Empirical Mutual Info.

Input: Dataset $\mathcal{D}$, target attribute α_t, k, p_f, ϵ
Output: An approximate top-k query answer

$C \leftarrow A \setminus \{\alpha_t\}$, $M \leftarrow M_0$, $i_{\max} \leftarrow \lceil \log_2 \frac{N}{M_0} \rceil + 1$, $p'_f \leftarrow \frac{p_f}{3i_{\max} \cdot (h-1)}$;
while $M \leq N$ **do**
  Calculate $\underline{H}(\alpha_t)$, $\overline{H}(\alpha_t)$, $b(\alpha_t)$ and λ by Lemma 3 with $p \leftarrow p'_f$;
  for $\alpha \in C$ **do**
    Calculate $\underline{I}(\alpha_t, \alpha)$, $\overline{I}(\alpha_t, \alpha)$, $b(\alpha)$ and $b(\alpha_t, \alpha)$;
    $b'(\alpha) \leftarrow b(\alpha_t) + b(\alpha) + b(\alpha_t, \alpha)$;
  $R \leftarrow$ top-k attributes from C according to $\overline{I}(\alpha_t, \alpha)$;
  $\overline{I}(\alpha_t, \alpha'_k) \leftarrow$ the k-th largest $\overline{I}(\alpha_t, \alpha)$ for $\alpha \in C$;
  $b'_{\max} \leftarrow$ the largest $b'(\alpha)$ for $\alpha \in R$;
  if $(\overline{I}(\alpha_t, \alpha'_k) - 6\lambda - b'_{\max})/\overline{I}(\alpha_t, \alpha'_k) \geq 1 - \epsilon$ **then**
    return R;
  else if $M < N$ **then**
    $M \leftarrow \min\{N, 2M\}$;
  else
    break;
  $\underline{I}(\alpha_t, \alpha''_k) \leftarrow$ the k-th largest $\underline{I}(\alpha_t, \alpha)$ for $\alpha \in C$;
  for $\alpha \in C$ **do**
    if $\overline{I}(\alpha_t, \alpha) < \underline{I}(\alpha_t, \alpha''_k)$ **then**
      $C \leftarrow C \setminus \{\alpha\}$;
return R;

where α'_k is the attribute with the k-th largest upper bound. If the stopping condition is met, then the algorithm terminates and returns an approximate answer. Otherwise, it doubles the sample size until all N records are sampled. It further prunes the attributes whose upper bound is smaller than the k-th largest lower bound (Algorithm 3 Lines 17-19).

We have the following theorem to state the correctness and time complexity of our top-k algorithm on empirical mutual information.

Theorem 5. *Algorithm 3 returns an approximate top-k answer satisfying Definition 5 with at least $1 - p_f$ probability. The expected running time can be bounded by:*

$$O\left(\min\left\{hN, \frac{h \log\left(h \log N / p_f\right) \log^2 N}{\epsilon^2 I^2\left(\alpha_t, \alpha_k^*\right)}\right\}\right).$$

The proof of Theorem 5 is omitted since it can follow similar steps as the proof of Theorems 1-2.

4.2 Approximate Filtering Query Processing

The filtering query algorithm on empirical mutual information is similar to that on empirical entropy. It samples M_0 in the beginning and adaptively doubles the sample size in each iteration until the stopping condition is met. The main difference is that p'_f is set to $\frac{p_f}{3i_{\max} \cdot (h-1)}$ since we only have $h-1$ possible attributes, and we

Algorithm 4: SWOPE-Filtering: Empirical Mutual Info.

Input: Dataset $\mathcal{D}$, target attribute α_t, p_f, ϵ, η
Output: An approximate filtering query answer

The steps are the same as Algorithm 2 except by changing $C = A \setminus \{\alpha_t\}$, $p'_f \leftarrow \frac{p_f}{3i_{\max} \cdot (h-1)}$, $\overline{H}$ to $\overline{I}$, $\underline{H}$ to $\underline{I}$, and $\hat{H}$ to $\hat{I}$;

derive the upper and lower bounds for $H(\alpha_t)$, $H(\alpha)$, $H(\alpha_t, \alpha)$ at most i_{max} times. The upper bounds, lower bounds, and estimated values for empirical entropy are changed to the upper bounds, lower bounds, and the estimated values for the empirical mutual information, respectively. The three cases in Algorithm 4 are then: *(i)* $\overline{I}(\alpha_t, \alpha) - \underline{I}(\alpha_t, \alpha) < 2\epsilon\eta$, *(ii)* $\underline{I}(\alpha_t, \alpha) \geq (1-\epsilon)\eta$, and *(iii)* $\overline{I}(\alpha_t, \alpha) < (1+\epsilon)\eta$. We have the following theorem for the correctness of Algorithm 4 and its time complexity.

Theorem 6. *Algorithm 4 returns an approximate filtering answer satisfying Definition 6 with at least $1 - p_f$ probability. The expected running time can be bounded by:*

$$O\left(\min\left\{hN, \frac{h \log\left(h \log N / p_f\right) \log^2 N}{\epsilon^2 \eta^2}\right\}\right).$$

The proof of Theorem 6 is also omitted since it can follow the similar steps as the proofs in Theorems 3-4.

5 THEORETICAL ANALYSIS

In this section, we present the detailed proofs of Theorem 2 and Theorem 4 and omit the proof of Lemma 3. The proof of Lemma 3 can be found in our technical report [1]. To prove Theorem 2 and 4, we introduce the following lemma.

Lemma 4. *When the sample size M is at least*

$$\frac{N\left(2 \log_2 N \sqrt{\frac{2 \ln(2/p) N}{N - 1/2}} + u_\alpha\right)^2}{(N-1)\kappa^2},$$

$2\lambda + b(\alpha) \leq \kappa$ holds with at least $1 - p/2$ probability, where λ (resp. b) is defined as Equation 6 (resp. 7) and κ is a positive real value.

Proof. According to the definition of λ in Equation 6,

$$\lambda = \left(\log_2 \frac{M}{M-1} + \frac{\log_2(M-1)}{M}\right) \sqrt{\frac{M(N-M)\ln(2/p)}{2(N-1/2)\left(1 - \frac{1}{2\max(M, N-M)}\right)}}.$$

Since $\log_2 \frac{M}{M-1} + \frac{\log_2(M-1)}{M} < \frac{2\log_2 M}{M}$ in [32] and $\max(M, N-M) \geq \frac{N}{2}$, we have $\lambda \leq \log_2 M \sqrt{\frac{2\ln(2/p) N(N-M)}{M(N-1/2)(N-1)}}$. Recall from Equation 7 that $b(\alpha) = \log_2\left(1 + \frac{(u_\alpha - 1)(N-M)}{M(N-1)}\right)$. Define $Z \in [0, 1]$ as $(N-M)/M/(N-1)$ and the range of Z comes from the fact that $1 \leq M \leq N$. To guarantee that $2\lambda + b(\alpha) \leq \kappa$, we ensure:

$$2 \log_2 M \sqrt{\frac{2 \ln(2/p) N}{N - 1/2}} \sqrt{Z} + \log_2(1 + (u_\alpha - 1) Z) \leq \kappa.$$

$\log_2 M$ has an upper bound $\log_2 N$ and $\log_2(1 + (u_\alpha - 1) Z) \leq (u_\alpha - 1) Z \leq u_\alpha Z \leq u_\alpha \sqrt{Z}$, where the last inequality comes from

the fact that $Z \le \sqrt{Z}$ for $Z \in [0,1]$. Then if we have that

$$\left(2\log_2 N\sqrt{\frac{2\ln(2/p)\,N}{N-1/2}}+u_\alpha\right)\sqrt{Z} \le \kappa,$$

$2\lambda + b(\alpha) \le \kappa$ will hold. After the transformation, we have that:

$$\frac{N-M}{M} \le (N-1)\,\kappa^2/(2\log_2 N\sqrt{\frac{2\ln(2/p)\,N}{N-1/2}}+u_\alpha)^2$$
$$\Leftrightarrow M \ge \frac{N}{\frac{(N-1)\kappa^2}{\left(2\log_2 N\sqrt{\frac{2\ln(2/p)N}{N-1/2}}+u_\alpha\right)^2}+1}.$$

Since we are deriving a lower bound for M, we can further discard the constant 1 in the denominator. We then require that

$$M \ge \frac{N\left(2\log_2 N\sqrt{\frac{2\ln(2/p)N}{N-1/2}}+u_\alpha\right)^2}{(N-1)\,\kappa^2},$$

which finishes the proof of the lemma. □

Proof of Theorem 2. The time complexity of estimating the lower and upper bound of $H(\alpha)$ with M samples is $O(M)$. Let c_0 be the constant factor of the time complexity of the estimation. Let M^* be the value of sample size M when Algorithm 1 terminates. There are h attributes in total. Then the expected running time of Algorithm 1 can be bounded by $c_0\,(h-1)\,M^*$.

Recall that the termination condition of Algorithm 1 is $(\overline{H}(\alpha'_k) - 2\lambda - b_{\max})/\overline{H}(\alpha'_k) \ge 1-\epsilon$. The left side of it is

$$(\overline{H}(\alpha'_k) - 2\lambda - b_{\max})/\overline{H}(\alpha'_k) = 1 - (2\lambda + b_{\max})/\overline{H}(\alpha'_k).$$

If $1-(2\lambda+b_{\max})/\overline{H}(\alpha'_k) \ge 1-\epsilon$, we have that:

$$(\overline{H}(\alpha'_k) - 2\lambda - b_{\max})/\overline{H}(\alpha'_k) \ge 1-\epsilon, \tag{8}$$

and Algorithm 1 will terminate. Equation 8 is equivalent to

$$2\lambda + b_{\max} \le \epsilon\overline{H}\left(\alpha'_k\right).$$

Recall that $\overline{H}(\alpha'_k) \ge H(\alpha^*_k)$, where $H(\alpha^*_k)$ is the exact k-th largest $H(\alpha)$ for $\alpha \in C$. Besides, $b_{\max}$ is the largest $b(\alpha)$ for $\alpha \in R$. If we can make sure that $2\lambda + b(\alpha) \le \epsilon H(\alpha^*_k)$, where α is the corresponding attribute with respect to $b_{\max}$, we have $2\lambda + b_{\max} \le \epsilon\overline{H}(\alpha'_k)$.

Replace κ with $\epsilon H(\alpha^*_k)$ and set p as p'_f in Lemma 4. When

$$M \ge \frac{N\left(2\log_2 N\sqrt{\frac{2\ln\left(2/p'_f\right)N}{N-1/2}}+u_\alpha\right)^2}{(N-1)\,\epsilon^2 H^2\left(\alpha^*_k\right)} \triangleq M^*,$$

then $2\lambda + b(\alpha) \le \epsilon H(\alpha^*_k)$ can be satisfied.

In Algorithm 1, the sample size M will double in each iteration and check whether M is large enough to satisfy the termination condition. So Algorithm 1 terminates with $M \le 2M^*$ with at least $1-p_f$ probability. Consider that the support size u_α is a constant in practice. Since $i_{\max} = \lceil\log_2(N/M_0)\rceil + 1$ and $p'_f = p_f/i_{\max}/h$ as we set in Algorithm 1, then we have that

$$M^* = O\left(\frac{\log\left(h\log N/p_f\right)\log^2 N}{\epsilon^2 H^2\left(\alpha^*_k\right)}\right).$$

The sample size M^* cannot exceed the number N and there are h candidate attributes in total. The expected running time of Algorithm 1 can be bounded by:

$$O\left(\min\left\{hN, \frac{h\log\left(h\log N/p_f\right)\log^2 N}{\epsilon^2 H^2\left(\alpha^*_k\right)}\right\}\right),$$

which finishes the proof. □

Table 2: Summary of datasets

Dataset	Rows	Columns
cdc-behavioral-risk	3753802	100
census-american-housing	14768919	107
census-american-population	31290943	179
enem	33714152	117

Proof of Theorem 4. In the approximate filtering query for empirical entropy, the most difficult case is when the estimated value of a empirical entropy is close to the preset threshold η. In this way, we do not prune this attribute until the difference between the upper and lower bound, i.e., $\overline{H}(\alpha) - \underline{H}(\alpha)$ is smaller than $2\epsilon\eta$. When $\overline{H}(\alpha) - \underline{H}(\alpha) < 2\epsilon\eta$, we are safe to return an approximate answer satisfying Definition 6.

For an attribute α which is not pruned until $\overline{H}(\alpha) - \underline{H}(\alpha) < 2\epsilon\eta$, we need to have $2\lambda + b(\alpha) < 2\epsilon\eta$ according to Lemma 3. We use $2\epsilon\eta$ to replace κ in Lemma 4 and p is set as p'_f. Then when

$$M \ge \frac{N\left(2\log_2 N\sqrt{\frac{2\ln\left(2/p'_f\right)N}{N-1/2}}+u_\alpha\right)^2}{4\,(N-1)\,\epsilon^2\eta^2} \triangleq M^*,$$

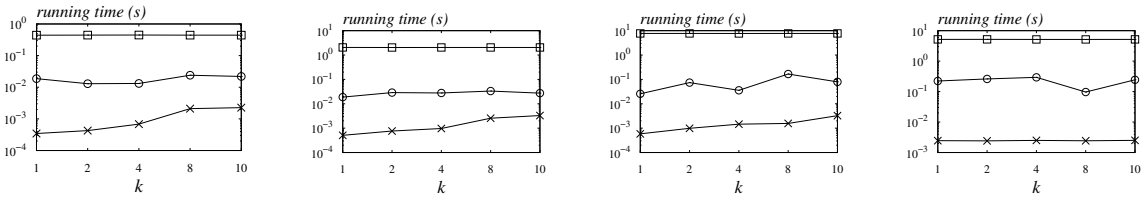
we have $2\lambda + b(\alpha) < 2\epsilon\eta$.

In our Algorithm 2, the sample size M will double in each iteration, and hence the sample size can be bounded by $2M^*$ with at least $1-p_f$ probability. The support size u_α can be regarded as a constant. Since $i_{\max} = \lceil\log_2(N/M_0)\rceil + 1$ and $p'_f = p_f/i_{\max}/h$ as we set in Algorithm 2, the above inequality is equal to

$$M^* = O\left(\frac{\log\left(h\log N/p_f\right)\log^2 N}{\epsilon^2\eta^2}\right)$$

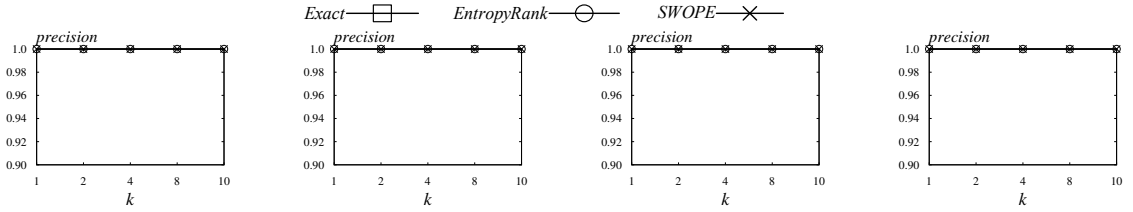
M cannot exceed the number of records N. Besides, there are h attributes. Then the expected running time of Algorithm 2 is:

$$O\left(\min\left\{hN, \frac{h\log\left(h\log N/p_f\right)\log^2 N}{\epsilon^2\eta^2}\right\}\right).$$



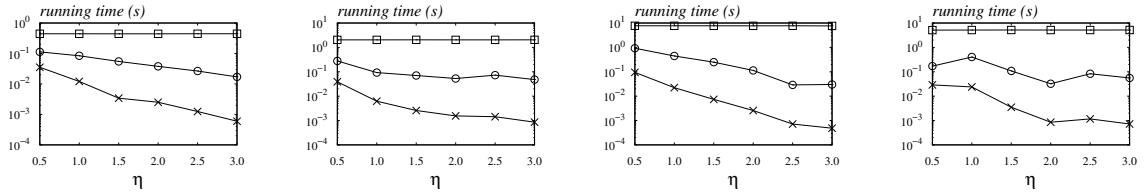
(a) cdc (b) hus (c) pus (d) enem

Figure 1: Varying k: Running time of empirical entropy top-k algorithms.



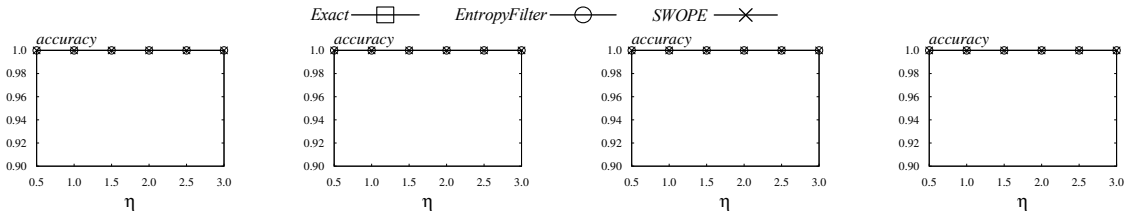
(a) cdc (b) hus (c) pus (d) enem

Figure 2: Varying k: Query precision of empirical entropy top-k algorithms.



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

Figure 3: Varying η: Running time of empirical entropy filtering algorithms.



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

Figure 4: Varying η: Query accuracy of empirical entropy filtering algorithms.

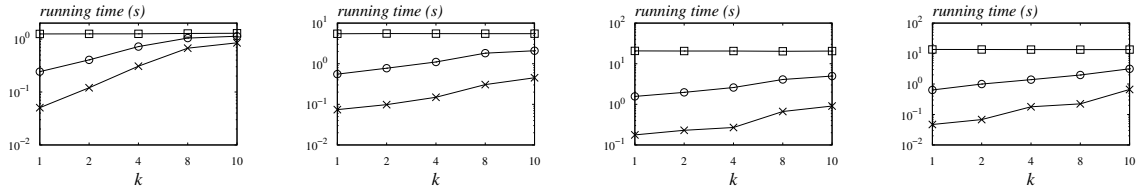
6 EXPERIMENTS

6.1 Experimental Settings

Datasets. We use four large real datasets: cdc-behavioral-risk (cdc), census-american-housing (hus), census-american-population (pus) and enem, that are publicly available and tested in [32]. The summary of these four datasets is shown in Table 2. Following [32], we remove columns with a too large support size, since they are usually not the preferred attributes for downstream data mining tasks. In our experiment, we eliminate columns with a support size larger than 1000. To create a test case for empirical mutual information, we choose one column as the target attribute and repeat the process for 20 times in each dataset. Each metric is averaged over 20 cases.

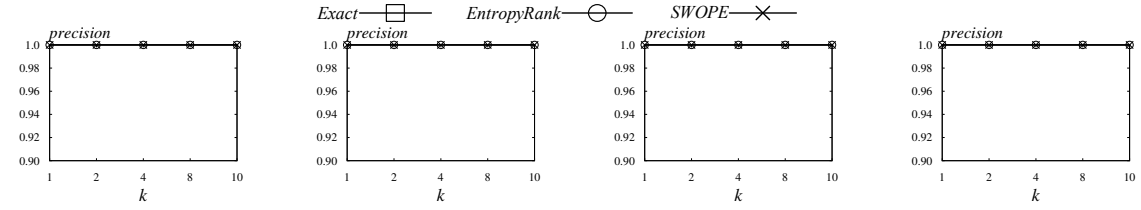
Algorithms. For empirical entropy queries, we compare our SWOPE top-k algorithm (resp. filtering algorithm) against the state-of-the-art top-k query algorithm (resp. filtering query algorithm) in [32], dubbed as *EntropyRank* (resp. *EntropyFilter*). We further include the exact solution, dubbed as *Exact*, as a baseline. For the empirical mutual information, we also include EntropyRank and EntropyFilter in [32] and the exact solution as our competitors. All algorithms are implemented with C++ and compiled with full optimization. All experiments are conducted on a Linux machine with an Intel Xeon 2.7GHz CPU and 200GB memory. Following [32], SWOPE stores data by columnar layout and do sequential sampling. To explain, random sampling on columnar layout may have a bad cache performance since it may randomly access different pages. This issue can be alleviated by sampling by the granularity of page sizes.

Parameter Settings. Recap that all our algorithms include a failure probability parameter p_f. We set $p_f = 1/N$ for our SWOPE, EntropyRank, and EntropyFilter. For top-k queries, we vary k in $[1, 10]$ following [32]. We show the results for k equal to 1, 2, 4, 8, 10. For the filtering queries, we still follow the setting in [32]. We vary η with $\{0.5, 1, 1.5, 2, 2.5, 3\}$ in empirical entropy filtering queries and vary η with $\{0.1, 0.2, 0.3, 0.4, 0.5\}$ in empirical mutual information filtering queries. We note that the settings of η are different since the empirical mutual information scores are typically smaller than the empirical entropy scores. Finally, SWOPE includes an error parameter ϵ to control the trade-off between the query accuracy and query efficiency. We tune the impact of ϵ in Section 6.4. The experiments show that when $\epsilon = 0.1$ (resp. $\epsilon = 0.05$), it achieves



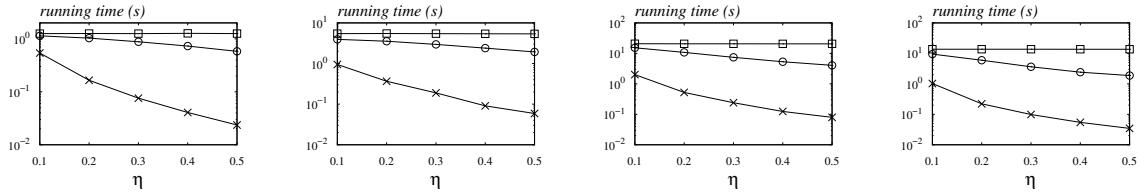
(a) cdc (b) hus (c) pus (d) enem

Figure 5: Varying k: Running time of empirical mutual information top-k algorithms.



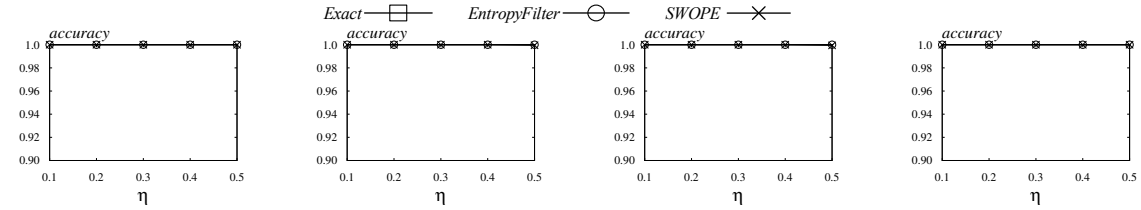
(a) cdc (b) hus (c) pus (d) enem

Figure 6: Varying k: Query accuracy of empirical mutual information top-k algorithms.



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

Figure 7: Varying η: Running time of empirical mutual information filtering algorithms.



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

Figure 8: Varying η: Query accuracy of empirical mutual information filtering algorithms.

the best trade-off between the query efficiency and accuracy on empirical entropy top-k queries (resp. filtering queries); when $\epsilon = 0.5$, it achieves the best trade-off between the query efficiency and accuracy on both the empirical mutual information top-k and filtering queries. In the rest of the experiments, we set these values of ϵ as default values in their corresponding top-k and filtering queries.

6.2 Evaluation on Empirical Entropy

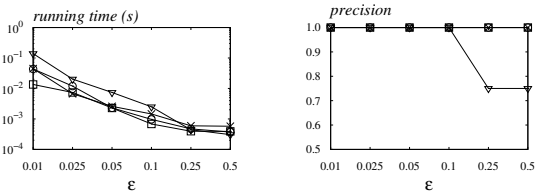
Top-k queries. In the first set of experiments, we evaluate the query performance and accuracy of the empirical entropy top-k queries on all four datasets by varying k from 1 to 10. Figure 1 shows the running time of our SWOPE against the competitors. As we can observe, our SWOPE consistently outperforms EntropyRank and is an order of magnitude faster than EntropyRank in most cases. Remarkably, our SWOPE is up to 117× faster than EntropyRank when $k = 4$ on the enem dataset. Compared with Exact, our solution further achieves up to three orders of magnitude improvement. In terms of accuracy, all three solutions provide the exact top-k answers achieving 100% accuracy in all cases.

Filtering queries. In the second set of experiments, we evaluate the query performance and accuracy of the filtering queries on all datasets by varying η from 0.5 to 3.0. Figure 3 reports the running time of all solutions. Again, our SWOPE is up to an order (resp. three orders) of magnitude faster than EntropyFilter (resp. Exact) in most cases. In particular, when $\eta = 3$, our SWOPE is 77× faster than EntropyFilter on enem dataset. In the meantime, our algorithm correctly reports all the attributes with empirical entropy no smaller than the threshold η in all cases as shown in Figure 4.

6.3 Evaluation on Empirical Mutual Info

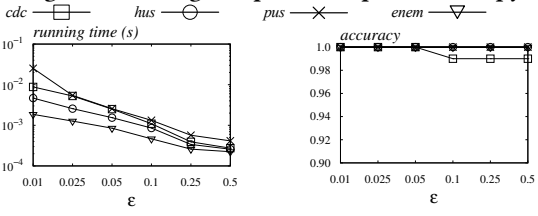
Top-k queries. Figure 5 reports the query time of all methods when we change k from 1 to 10. As we can observe, SWOPE is still the most efficient algorithm among all methods: SWOPE is up to an order (resp. two orders) of magnitude faster than EntropyRank (resp. Exact). In terms of accuracy, all the methods still provide identically highly accurate results as shown in Figure 6.

Filtering queries. Figure 7 reports the query time of all methods when we change η from 0.1 to 0.5. SWOPE is up to 54× faster than EntropyFilter and two orders of magnitude faster than Exact. In the meantime, SWOPE reports identically accurate results as EntropyFilter and Exact.



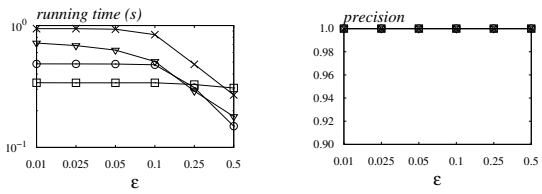
(a) Running Time (b) Precision

Figure 9: Tuning ϵ: top-k on empirical entropy.



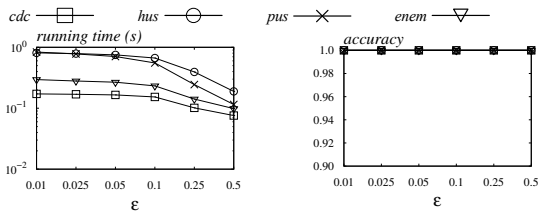
(a) Running Time (b) Accuracy

Figure 10: Tuning ϵ: filtering on empirical entropy.



(a) Running Time (b) Precision

Figure 11: Tuning ϵ: top-k on empirical mutual info.



(a) Running Time (b) Accuracy

Figure 12: Tuning ϵ: filtering on empirical mutual info.

6.4 Tuning ϵ

Finally, we examine the trade-off between the query efficiency and accuracy of our approximate algorithms by varying ϵ with $\{0.01, 0.025, 0.05, 0.1, 0.25, 0.5\}$ for the above four queries on all datasets. We fix $k = 4$ for both empirical entropy and mutual information top-k queries. We fix $\eta = 2$ (resp. $\eta = 0.3$) for empirical entropy (resp. mutual information) filtering query. As shown in Figures 9-12, when we increase ϵ, the running time of all algorithms decreases. However, on different queries, ϵ further impacts the accuracy. As shown in Figure 9(b), on the empirical entropy top-k queries, when ϵ increases from 0.1 to 0.25, the accuracy decreases from 100% to around 75%. Therefore, we choose $\epsilon = 0.1$ as the default value for the empirical entropy top-k queries. According to Figure 10(b), on empirical entropy filtering queries, when ϵ increases from 0.05 to 0.1, the accuracy changes from 1 to 0.99. Therefore, we choose $\epsilon = 0.05$ as the default value for empirical entropy filtering queries. Finally, as shown in Figures 11(b) and 12(b), in all settings of ϵ, both the empirical mutual information top-k and filtering queries achieve 100% accuracy. Therefore, we set $\epsilon = 0.5$ as the default value for both the empirical mutual information top-k and filtering queries.

7 RELATED WORK

Entropy stands as a fundamental concept in data mining and information theory [28]. A plethora of research work has focused on developing efficient algorithms to estimate the information entropy, e.g., [17, 18, 25, 30, 38]. Paninski [25] finds the connection between the bias of entropy estimators and a certain polynomial approximation problem. He then develops an estimator equipped with rigorous bounds on the maximum error over all possible underlying probability distributions. This also bounds the difference between empirical entropy and information entropy. Valiant et al. [30] derive that given a sample of independent draws from any distribution over at most u distinct elements, the entropy can be estimated using $O(u/\log u)$ samples. Jiao et al. [17] and Wu et al. [38] design algorithms to estimate the entropy with minimax squared error rate $c^2/(M \ln M)^2 + \ln^2 u/M$ using the polynomial approximation based on M samples where $M \gg u/\ln u$. Besides, Jiao et al. [18] shows an estimation of the entropy with $u^2/M^2 + \ln^2 u/M$ worst case square error rates of MLE using $M \gg u$ samples. There also exist studies on entropy monitoring under streaming or distributed settings, e.g., [7, 15]. However, none of these research works consider the top-k or filtering queries under empirical entropy or mutual information. Most recently, Wang et al. [32] adopts the sampling without replacement techniques to answer the top-k and filtering queries as we discuss in Section 2. However, their solution still targets to return the exact answer and leaves much room for improvement.

Besides, a plethora work focuses on approximate top-k queries, e.g., [6, 29, 36]. Sheng et al. [29] propose an ϵ-approximate algorithm to find k vertices with the largest degrees on a hidden bipartite graph. Cao et al. [6] discuss how to choose k distributions with the largest means efficiently tolerating a small relative error. Wang et al. [36] consider the approximate top-k personalized PageRank queries. There also exists a line of research work, e.g., [8, 9], focusing on heavy hitter queries that share a similar spirit as our filtering queries. In heavy hitter queries, we are given a stream of length N, and it asks for the elements with a frequency that is larger than $\tau \cdot N$, where τ is a threshold. All of these queries can be regarded as the sum or average of a random variable, and therefore classic concentration bound can be applied. However, estimating empirical entropy is more challenging, and this motivates us to design more efficient and effective approximate top-k and filtering query algorithms.

8 CONCLUSION

In this paper, we present an efficient framework SWOPE to handle the top-k and filtering queries on empirical entropy and mutual information. Theoretical analysis shows that our proposed solution achieves improved time complexity than existing alternatives. Experiments show that our solution is up to two orders of magnitude faster than alternatives without sacrificing the query accuracy.

ACKNOWLEDGMENTS

This work is supported by Hong Kong RGC ECS (No. 24203419), Hong Kong RGC CRF (No. C4158-20G), CUHK Direct Grant (No. 4055114), and NSFC (No. U1936205).

REFERENCES

[1] 2020. Technical Report. https://www.dropbox.com/sh/ykli4ym6koh8h7v/AADuTh-I9RGI5sWuwFfDQyMsa?dl=0.

[2] Fatemeh Amiri, Mohammad Mahdi Rezaei Yousefi, Caro Lucas, Azadeh Shakery, and Nasser Yazdani. 2011. Mutual information-based feature selection for intrusion detection systems. *J. Netw. Comput. Appl.* 34, 4 (2011), 1184–1199.

[3] Bădulescu Laviniu Aurelian. 2018. An information entropy based splitting criterion better for the Data Mining Decision Tree algorithms. In *ICSTCC*. 535–540.

[4] Daniel Barbará, Yi Li, and Julia Couto. 2002. COOLCAT: an entropy-based algorithm for categorical clustering. In *CIKM*. 582–589.

[5] Boyan Bonev, Francisco Escolano, and Miguel Cazorla. 2008. Feature selection, mutual information, and the classification of high-dimensional patterns. *Pattern Anal. Appl.* 11, 3-4 (2008), 309–319.

[6] Wei Cao, Jian Li, Yufei Tao, and Zhize Li. 2015. On Top-k Selection in Multi-Armed Bandits and Hidden Bipartite Graphs. In *NeurIPS*. 1036–1044.

[7] Graham Cormode. 2013. The continuous distributed monitoring model. *SIGMOD Rec.* 42, 1 (2013), 5–14.

[8] Graham Cormode and Marios Hadjieleftheriou. 2009. Finding the frequent items in streams of data. *Commun. ACM* 52, 10 (2009), 97–105.

[9] Graham Cormode and S. Muthukrishnan. 2005. An improved data stream summary: the count-min sketch and its applications. *J. Algorithms* 55, 1 (2005), 58–75.

[10] Luis M. de Campos. 2006. A Scoring Function for Learning Bayesian Networks based on Mutual Information and Conditional Independence Tests. *J. Mach. Learn. Res.* 7 (2006), 2149–2187.

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

[12] Pablo A. Estévez, M. Tesmer, Claudio A. Perez, and Jacek M. Zurada. 2009. Normalized Mutual Information Feature Selection. *IEEE Trans. Neural Networks* 20, 2 (2009), 189–201.

[13] François Fleuret. 2004. Fast Binary Feature Selection with Conditional Mutual Information. *J. Mach. Learn. Res.* 5 (2004), 1531–1555.

[14] Pawel Foremski, David Plonka, and Arthur W. Berger. 2016. Entropy/IP: Uncovering Structure in IPv6 Addresses. In *IMC*. 167–181.

[15] Moshe Gabel, Daniel Keren, and Assaf Schuster. 2017. Anarchists, Unite: Practical Entropy Approximation for Distributed Streams. In *SIGKDD*. 837–846.

[16] Yanjun Han, Jiantao Jiao, and Tsachy Weissman. 2015. Adaptive estimation of Shannon entropy. In *ISIT*. 1372–1376.

[17] Jiantao Jiao, Kartik Venkat, Yanjun Han, and Tsachy Weissman. 2015. Minimax Estimation of Functionals of Discrete Distributions. *IEEE Trans. Inf. Theory* 61, 5 (2015), 2835–2885.

[18] Jiantao Jiao, Kartik Venkat, Yanjun Han, and Tsachy Weissman. 2017. Maximum Likelihood Estimation of Functionals of Discrete Distributions. *IEEE Trans. Inf. Theory* 63, 10 (2017), 6774–6798.

[19] Ambika Kaul, Saket Maheshwary, and Vikram Pudi. 2017. AutoLearn - Automated Feature Generation and Selection. In *ICDM*. 217–226.

[20] 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.

[21] Tao Li, Sheng Ma, and Mitsunori Ogihara. 2004. Entropy-based criterion in categorical clustering. In *ICML*, Carla E. Brodley (Ed.), Vol. 69.

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

[23] Rajeev Motwani and Prabhakar Raghavan. 1995. *Randomized algorithms*. Cambridge university press.

[24] Jaganathan Palanichamy and Kuppuchamy Ramasamy. 2013. A threshold fuzzy entropy based feature selection for medical database classification. *Comput. Biol. Medicine* 43, 12 (2013), 2222–2229.

[25] Liam Paninski. 2003. Estimation of Entropy and Mutual Information. *Neural Computation* 15, 6 (2003), 1191–1253.

[26] Hanchuan Peng, Fuhui Long, and Chris H. Q. Ding. 2005. Feature Selection Based on Mutual Information: Criteria of Max-Dependency, Max-Relevance, and Min-Redundancy. *IEEE Trans. Pattern Anal. Mach. Intell.* 27, 8 (2005), 1226–1238.

[27] J. Ross Quinlan. 1986. Induction of Decision Trees. *Mach. Learn.* 1, 1 (1986), 81–106.

[28] Claude E Shannon. 1948. A mathematical theory of communication. *The Bell system technical journal* 27, 3 (1948), 379–423.

[29] Cheng Sheng, Yufei Tao, and Jianzhong Li. 2012. Exact and approximate algorithms for the most connected vertex problem. *ACM Trans. Database Syst.* 37, 2 (2012), 12:1–12:39.

[30] Paul Valiant and Gregory Valiant. 2013. Estimating the Unseen: Improved Estimators for Entropy and other Properties. In *NeurIPS*. 2157–2165.

[31] Jorge R. Vergara and Pablo A. Estévez. 2014. A review of feature selection methods based on mutual information. *Neural Computing and Applications* 24, 1 (2014), 175–186.

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

[33] Qing Ren Wang and Ching Y. Suen. 1984. Analysis and Design of a Decision Tree Based on Entropy Reduction and Its Application to Large Character Set Recognition. *IEEE Trans. Pattern Anal. Mach. Intell.* 6, 4 (1984), 406–417.

[34] 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.

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

[36] 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.

[37] 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.

[38] Yihong Wu and Pengkun Yang. 2016. Minimax Rates of Entropy Estimation on Large Alphabets via Best Polynomial Approximation. *IEEE Trans. Inf. Theory* 62, 6 (2016), 3702–3720.

[39] Yiming Yang and Jan O. Pedersen. 1997. A Comparative Study on Feature Selection in Text Categorization. In *ICML*, Douglas H. Fisher (Ed.). Morgan Kaufmann, 412–420.