Interpretable and Informative Log Summarization

[Authors Anonymized]
[Institution Anonymized]
[email anonymized]

ABSTRACT

Analyzing database access logs is a key part of performance tuning, intrusion detection, benchmark development, and many other database administration tasks. Unfortunately, it is common for production databases to deal with millions or even more queries each day, and so logs must be summarized for human consumption. Designing an appropriate summary requires trading off between conciseness and information content. In this paper we formalize and analyze this tradeoff in the context of two families of "pattern-based" and "pattern-mixture" log summaries; We precisely characterize the information content of a log, the verbosity of a summary, and define two measures of information loss due to summarization: ambiguity and deviation. As neither ambiguity nor deviation are efficiently computable, we define an approximation that tracks both ambiguity and deviation: summary error. We then propose a restricted class of "naive pattern-mixture" summaries that admit an efficient algorithm for generating summaries that achieve specific tradeoffs between verbosity and error. Experiments performed on a prototype implementation of this algorithm show that, compared to a state of the art summarization technique called LaserLight, pattern-mixture summaries are both faster to create and more informative.

1. INTRODUCTION

Database access logs are used in a wide variety of settings, including evaluating database performance tuning [5], benchmark development [25], database auditing [29], compliance validation [11] and query recommendation [8]. Many of these cases — benchmark development and database auditing in particular — require being able to explain patterns and outliers in the log, something that as yet can not be fully automated. Although such tasks benefit from manual log analysis by human experts, access logs can grow to be extremely large. For instance, a recent study of queries at a major US bank for a period of 19 hours found nearly 17 million SQL queries and over 60 million stored procedure execution events [29]. Numerous approaches to log summarization [34, 35, 39, 19, 15, 3, 38, 23, 41, 13] promise to make manual analysis more practical by helping analysts to quickly identify common patterns and outliers.

Nearly all existing log summarization techniques rely on underlying measures of inter-query similarity. For example, one approach [1] measures similarity by the degree of overlap between structural elements like `SELECT` items, `FROM` tables, or conjunctive clauses in the `WHERE` clause. Unfortunately, query similarity is usually context-dependent. For example, two queries that are similar with respect to performance may have very different impacts on a security audit. Indeed, to the best our knowledge, similarity metrics for log summarization are all defined heuristically in terms of a fixed set of relationships that target specific log analysis tasks. As a result, similarity-based summarization techniques are not easy to generalize to new settings.

In this paper, we consider the challenge of log summarization from a task-agnostic, information theoretical perspective: *How can we communicate the most detail about the log with as simple a description as possible?* To accomplish this, we first decouple the process of feature engineering (i.e., of identifying features relevant to a given task) from the process of summarizing feature-based representations of the log. We then define a family of *pattern-based* summaries, as well as a more general family of *pattern-mixture* summaries. These summary families give us a framework for reasoning about both the complexity and descriptive power of a summary in a principled way. Concretely, we define three measures over both families: (1) Verboseness, which measures the complexity of the summary; (2) Ambiguity, which measures the precision with which a summary captures the log that generated it; and (3) Deviation, which measures how misleading the summary is as a result of missing details. Neither Ambiguity nor Deviation can be computed efficiently, so we propose a fourth measure called *Summary Error*. We show through a combination of theoretical proofs and experimental validation that Summary Error is a reliable predictor of both Ambiguity and Deviation.

In general, Verbosity and Summary Error are inversely related: The more detailed the summary, the more precisely it captures the original log. Thus, log summarization may be defined as a search over a space of summaries to identify the summary that best trades off between these two properties. Unfortunately, searching for such an ideal summary from the full space of pattern-based summaries is computationally infeasible. To avoid this limitation, we restrict our search to a family of *naive mixture summaries* — a sim-

plified form of pattern mixture summaries. We show how to efficiently construct such a summary by first clustering entries in the log and then summarizing each cluster separately. This approach may then be further refined: (1) By *hierarchically sub-clustering* the data to scrub through different trade-offs between verboseness and summary error, and (2) By *piggybacking* on more complex, state-of-the-art pattern-based summarization approaches to create more intricate summaries.

Concretely, in this paper we make the following contributions: (1) We define two families of summary visualizations: pattern-based and pattern-mixture, (2) We define Verbosity, Ambiguity, and Deviation, three principled measures of the quality of a pattern-based or pattern-mixture summary, (3) We define the computationally efficient measure, Summary Error, and demonstrate that it is a close approximation of both Ambiguity and Deviation, (4) We propose using clustering to generate a family of naive mixture summaries and experimentally show how to efficiently optimize for any given tradeoff between Summary Error and Verbosity within this family through structure-wise sub-clustering, and (5) We experimentally show how to extend naive mixture summaries content-wise by piggybacking state-of-the-art pattern-based summarization approaches, and experimentally measure the difference in the ratio of Summary Error versus Verbosity and also running time.

2. BACKGROUND AND RELATED WORK

Existing approaches for log summarization are frequently aimed at specific tasks like query recommendation [8, 14, 27, 40], performance optimization [2, 6], session identification [1], outlier detection [24], or workload analysis [32].

Chatzopoulou *et al.* [8] aim to assist non-expert users of scientific databases by tracking their historical querying behavior and generating personalized query recommendations. They flatten query AST as a bag of *fragments* and adopt *feature vector* representation of queries. User profiles are then built from the query log by summarizing feature vectors that belong to workloads of the same user. Similarly, Giacometti *et al.* [14] aim at making query recommendation by summarizing queries in the previous sessions. SnipSuggest [27] is a context-aware SQL-autocomplete system that helps database users to write SQL queries piece by piece, by suggesting SQL *snippets*. It computes the marginal probability that a query, uniformly drawn from the log, contains a snippet. Snippets and their marginals serving as the summary of the query log, which assists the prediction on the snippet that a user will most likely to append to existing snippets. Yang *et al.* [40] also aim at assisting users in writing SQL queries. They build a graph for each query in the log using tables in *join* operation and cluster queries into similarity groups based on these graphs.

Aouiche *et al.* [2] aim to help databases respond to users' queries faster through optimizing view selection in warehouses by summarizing query logs. They consider operations of *selection*, *joins* and *group-by* in the query to create feature vectors. Bruno *et al.* [6] aim at summarizing multi-dimensional data tuples stored in database relations for selectivity estimation during query optimization and approximate query processing. The summary is represented as *multi-dimensional histograms*.

Aligon *et al.* [1] study various approaches on identifying similar OLAP sessions for the purpose of query recommendation and personalization. They identify sub-trees in query abstract syntax tree that represent operations of *selection* and *join* as the most relevant in a query followed by the *group by*.

Kamra *et al.* [24] aim at detecting anomalous behavior of queries in the log by summarizing query logs into profiles of normal user behavior interacting with a database.

Makiyama *et al.* [32] approach query log summarization with the goal of analyzing a system's workload, and they provide a set of experiments on Sloan Digital Sky Survey (SDSS) dataset. They extract features from query AST by considering operators *selection*, *joins*, *projection*, *from*, *group-by* and *order-by* separately.

There are also works that aim at summarizing a more general data representation—multi-dimensional vectors where attributes are either discrete or binary. Gebaly *et al.* [12] aim at summarizing multi-dimensional discrete-valued vectors augmented with a binary attribute. The summary is represented as a collection of patterns. Mampaey *et al.* [33] study the problem of summarizing multi-dimensional vectors where all attributes are binary-valued. The summary is represented as a set of *most informative* patterns which maximize the *Bayesian Information Criterion*(BIC) score.

Works related to query log summarization and more generally, multi-dimensional vector summarization are not limited to ones described above. In addition, there are studies that focus on visualization/interpretability of query log summarization.

QueryViz [9] addresses *query interpretation* which is the problem of understanding the goal of the query by visualizing it graphically. QueryScope [20] aims at finding better tuning opportunities by helping human experts to visualize and identify patterns shared among queries. Logos [28], on the other hand, is a system that has the ability to translate SQL queries into natural language equivalents.

3. PATTERN-BASED SUMMARIES

In this section, we formally define the problem of log summarization from a task-agnostic, information theoretical perspective.

3.1 Notation

We define a database log to be an order-free collection of queries, each defined as a collection of features. For ease of exposition, we adopt the conventions of Aligon et. al. [1], assume conjunctive queries, and define queries in terms of the following three types of features: (1) tables in the `FROM` clause, (2) columns in the `SELECT` clause, and (3) atomic boolean expressions in the `WHERE` clause. We use f to denote the feature (table, column, or atomic expression), and C to denote the feature's category (`FROM`, `SELECT`, or `WHERE`). We refer to the 2-tuple $w = \langle f, C \rangle$ as a *word*.

EXAMPLE 1. *Consider the following example query.*

```
SELECT _id, sms_type, _time FROM Messages
WHERE status = 1 AND transport_type = 3
```

This query uses 6 words: $\langle$ `sms_type`, `SELECT` $\rangle$, $\langle$ `_id`, `SELECT` $\rangle$, $\langle$ `_time`, `SELECT` $\rangle$, $\langle$ `Messages`, `FROM` $\rangle$, $\langle$ `status = 1`, `WHERE` $\rangle$, *and* $\langle$ `transport_type = 3`, `WHERE` $\rangle$

Denote by $\mathbf{b}$ an arbitrary *bag of words*, encoded as a vector $\mathbf{b} = (x_1, \ldots, x_n)$ where n is the number of distinct words and

$x_i = k$ indicates ith labeled word w_i occurs k times in the bag. For any two bags $\mathbf{b}$, $\mathbf{b}'$, we say that $\mathbf{b}'$ *is contained* in $\mathbf{b}$, denoted $\mathbf{b}' \subseteq \mathbf{b}$, if $\mathbf{b}'$ contains a subset of the words in $\mathbf{b}$:

$$\mathbf{b}' \subseteq \mathbf{b} \quad \equiv \quad \forall i,\, x'_i \leq x_i$$

When it is clear from context, we will use features and words interchangeably. Observe that the bag representation and original SQL representations are homomorphic modulo schema reordering and query equivalence — It is possible to transform any bag of words into a corresponding conjunctive query, and visa versa[1]. We thus abuse the notation $\mathbf{q}$ to denote both a conjunctive query and its corresponding bag of words encoding.

3.2 Summarizing the Log

We would like to be able to effectively communicate the information content of a query log in an interpretable way. Displaying the entire log communicates this information completely, but is a large amount of information for a user to digest. Thus, our immediate goal is a summary representation that communicates the log's information content, but in a more compact form. We start by formally defining the information content of the log as a whole, and then use this definition to create a generic model for summaries.

Information Content of Logs. Denote by L a query log, a bag of queries $\{\ \mathbf{q} \mid \mathbf{q} \in L\ \}$. We define the information content of the log as the distribution $p(Q \mid L)$ of queries uniformly drawn from the log.

EXAMPLE 2. *Consider the following log query log, which consists of five queries.*

```
1. SELECT _id FROM Messages WHERE status = 1
2. SELECT _time FROM Messages
          WHERE status = 1 AND sms_type = 1
3. SELECT _id FROM Messages WHERE status = 1
4. SELECT _id, _time FROM Messages
          WHERE sms_type = 1
5. SELECT sms_type, _time FROM Messages
          WHERE sms_type = 1
```

Drawing uniformly from the log, each entry will appear with probability $\frac{1}{5} = 0.2$. *The following query occurs twice:*

$\mathbf{q}_1 = \mathbf{q}_3 =$ `SELECT _id FROM Messages WHERE status = 1`

Hence the probability of drawing $\mathbf{q}_1$ *is double that of the others (i.e.,* $p(Q = \mathbf{q}_1 \mid L) = p(Q = \mathbf{q}_3 \mid L) = \frac{2}{5} = 0.4$*)*

Decomposing the query into its component words, we can define a specific query $\mathbf{q} = (x_1, \ldots, x_n)$ to be an observation of the multivariate distribution over variables $Q = (X_1, \ldots, X_n)$ with probability:

$$p(Q = \mathbf{q} \mid L) = \frac{|\{\!|\ \mathbf{q}' \mid \mathbf{q}' = \mathbf{q} \wedge \mathbf{q}' \in L\ |\!\}|}{|L|}$$

EXAMPLE 3. *Continuing the example, the log's vocabulary consists of (1)* $\langle$ `_id, SELECT` $\rangle$, *(2)* $\langle$ `_time, SELECT` $\rangle$, *(3)* $\langle$ `sms_type, SELECT` $\rangle$, *(4)* $\langle$ `status = 1, WHERE` $\rangle$, *(5)* $\langle$ `sms_type = 1, WHERE` $\rangle$, *and (6)* $\langle$ `Messages, FROM` $\rangle$. *Accordingly, the queries can be encoded as 6-tuples, with fields counting the number of occurrences of each word:* $\mathbf{q}_1 = \langle\ 1, 0, 0, 1, 1, 0\ \rangle$, $\mathbf{q}_2 = \langle\ 0, 1, 0, 1, 1, 1\ \rangle$, $\mathbf{q}_3 = \langle\ 1, 0, 0, 1, 1, 0\ \rangle$, $\mathbf{q}_4 = \langle\ 1, 1, 0, 1, 0, 1\ \rangle$, $\mathbf{q}_5 = \langle\ 0, 1, 1, 1, 0, 1\ \rangle$

[1]In general, the specific process for converting queries to collections of features is irrelevant. We only require only that summaries can be generated from bags of features as in Aligon.

SELECT	sms_type, external_ids, _time, _id
FROM	messages
WHERE	(sms_type=1) ∧ (sms_type=0) ∧ (status=1) ∧ (_time≥1426084288402000) ∧ (transport_type=3)

(a) *Correlation-ignorant*: Words are highlighted independently

SELECT	sms_type
FROM	messages
WHERE	sms_type=1

SELECT	sms_type
FROM	messages
WHERE	status=1

(b) *Correlation-aware*: Groups of words (patterns) are highlighted together, according to the probability of co-occurrence.

Figure 1: **Example pattern-based summaries: Darker elements are more likely to occur in the log.**

Our immediate goal can thus be restated as summarizing the multivariate distribution $p(Q)$.

Use Cases of Information Content. Before we model the summary representation of the information content, we offer three example use cases:

Index Selection: Words of atomic boolean expressions that frequently occur in the query log are good candidates for indexing. e.g. word `status = 1` occurs 90% of the time in queries drawn from the log. This suggests indexing column `status`.

Materializing Views Selection: Words of atomic boolean expressions that frequently co-occur with a specific table are good candidates for materialization. e.g. words `status = 1` and `sms_type = 1` co-occur 90% of the time with table `Messages`. This suggests materializing the view that filters table `Messages` by condition `status = 1 AND sms_type = 1`.

Detecting Outliers: Two similar sets of co-occurring words but with significantly different frequencies could indicate a bug or malicious use of the system [30]. e.g. words `SELECT balance` and `FROM accounts` co-occur with `password=?` 90% of the time. This implies that queries containing these words are used for validating the password from user input. Suppose `SELECT balance` and `FROM accounts` also co-occur with tautology `1=1`, 1% of the time. This suggests someone is trying to bypass the password checking.

The naive way to approach the problem is to simply describe the frequency of each word occurring in the log, for example by using shading as in Figure 1a. In the case of index selection, such a representation might be sufficient — frequently occurring constraints or attributes are good candidates for indexing. However, in the other two examples the user is not interested in occurrence, but rather co-occurrence. In other words, a viable summary needs to capture correlations in the data as well. For example, Figure 1b illustrates a specific visual representation of correlations in the log. The shading of correlated words to the right clearly communicates that words `SMS_TYPE`, `MESSAGES`, and `STATUS=1` do not co-occur as frequently as it would appear from Figure 1a.

Pattern-based Summaries. Both visual encodings in Figure 1 belong to a broader class of what we call *pattern-based summaries*, that are able to communicate correlations

through co-occurrence probabilities. Concretely, a *pattern* is an arbitrary bag of words $\mathbf{b}$ that can occur/co-occur in a query drawn from the query log. Each pattern conveys a piece of information on distribution $p(Q)$ through the probability of uniformly drawing a query $\mathbf{q}$ from the log that *contains* the pattern (i.e., $\mathbf{q} \supseteq \mathbf{b}$). This probability is equivalent to the marginal $p(X_1 \geq x_1, \ldots, X_n \geq x_n)$[2] where $\mathbf{b} = (x_1, \ldots, x_n)$. We denote this probability as:

$$p(Q \supseteq \mathbf{b}) \equiv \frac{|\{\!|\ \mathbf{q} \mid\ \mathbf{q} \supseteq \mathbf{b} \wedge \mathbf{q} \in L\ |\!\}|}{|L|} = \sum_{\forall \mathbf{q} \in L \wedge \mathbf{q} \supseteq \mathbf{b}} p(Q = \mathbf{q} \mid L)$$

Denote by $S_{max} : \mathbb{N}^n \to [0, 1]$, the full mapping from the space of all possible patterns $\mathbf{b} \in \mathbb{N}^n$ to their marginal probabilities. A pattern-based summary is then a visual encoding that can be expressed as a partial mapping $S \subseteq S_{max}$. We use $S[\mathbf{b}]$ to represent the marginal mapped from $\mathbf{b}$. When it is clear from context, we abuse syntax and also use S to denote its set of mapped patterns (i.e., $dom(S)$). Hence, $|S|$ is the number of mapped patterns, which we call the *verbosity* of the summary.

Note that both visual encodings in Figure 1 are pattern-based summaries: (a) the correlation-ignorant summary consists of 10 patterns, one for each word and (b) the correlation-aware summary consists of two patterns (left and right). Henceforth, we will assume that we already have an user-interface for visualizing pattern-based summaries and focus on selecting which specific summary to present.

4. ANALYZING SUMMARIES

Ultimately, our goal is to effectively communicate the distribution $p(Q)$ to the user. On the one hand, less complex summaries (i.e., those with fewer patterns) are desirable. On the other hand, we also want to ensure that the summary is representative of the distribution $p(Q)$ being communicated. We will return to the tradeoff between representativeness and complexity in Section 6. However, first we need to define what it means for a summary to be representative.

4.1 Lossless Summaries

To establish a baseline for measuring representativeness, we begin with the extreme cases. At one extreme, an empty summary ($|S| = 0$) conveys no information. At the other extreme, we have the summary S_{max} which is the full mapping from all patterns. Having this summary is a sufficient condition to reconstruct the original distribution $p(Q)$.

PROPOSITION 1. *For any query $\mathbf{q} = (x_1, \ldots, x_n) \in \mathbb{N}^n$, the probability of drawing exactly $\mathbf{q}$ at random from the log (i.e., $p(X_1 = x_1, \ldots, X_n = x_n)$) is computable, given S_{max}.*

PROOF. Denote by $\mathbb{1}^n = \{0, 1\}^n$ the space of possible 0-1 vectors of size n, and define a summary $\overline{S}_{\mathbf{q}}$ with patterns:

$$dom(\overline{S}_{\mathbf{q}}) = \{\ (x_1 + b_1, \ldots, x_n + b_n) \mid\ (b_1, \ldots, b_n) \in \mathbb{1}^n\ \}$$

We will show that $\overline{S}_{\mathbf{q}} \subseteq S_{max}$ contains sufficient information to compute $p_0 = p(X_1 = x_1, \ldots, X_n = x_n)$ through several steps. First, we define a new pair of marginal probabilities $p_1 \langle\ b_1\ \rangle = p(X_1 \geq x_1 + b_1, X_2 = x_2, \ldots, X_n = x_n)$. x_1 is integral, so $p_0 = p_1 \langle\ 0\ \rangle - p_1 \langle\ 1\ \rangle$. Generalizing, we can define:

$$p_k \langle\ b_1, \ldots, b_k\ \rangle = p(X_1 \geq x_1 + b_1,\ \ldots,\ X_k \geq x_k + b_k,\ X_{k+1} = x_{k+1},\ \ldots,\ X_n = x_n)$$

Again, x_k being integral gives us that:

$$p_{k-1} \langle\ b_1, \ldots, b_{k-1}\ \rangle = p_k \langle\ b_1, \ldots, b_{k-1}, 0\ \rangle - p_k \langle\ b_1, \ldots, b_{k-1}, 1\ \rangle$$

Finally, when $k = n$, the probability $p_n \langle\ b_1, \ldots, b_n\ \rangle$ is the marginal probability $p(Q \supseteq \mathbf{b})$ of a pattern $\mathbf{b} = (x_1 + b_1, \ldots, x_n + b_n)$, which by definition is offered by $\overline{S}_{\mathbf{q}}$ for any $(b_1, \ldots, b_n) \in \mathbb{1}^n$. □

The resulting summary $\overline{S} = \bigcup_{\mathbf{q} \in L} \overline{S}_{\mathbf{q}}$ fully reconstructs the distribution $p(Q)$. We refer to any summary that can fully reconstruct distribution $p(Q)$, a *lossless* summary. Clearly any summary that extends $\overline{S}$ (including S_{max}) is lossless.

Unfortunately, while lossless summaries are precise, they are also verbose. At the extreme, S_{max} is tantamount to communicating the entire log. Hence, for the remainder of the paper, we focus on lossy summaries.

4.2 Lossy Summaries

A lossy summary $S \subset S_{max}$ may not be able to precisely identify the distribution $p(Q)$, but still communicates something about its information content. We characterize the information content of a lossy summary S by defining a *space* (denoted Ω_S) of possible distributions $\rho \in \Omega_S$ allowed by a summary S. This space is defined by a set of constraints as follows. First, we have the general properties of probability distributions:

$$\forall \mathbf{q} \in \mathbb{N}^n : \rho(\mathbf{q}) \geq 0 \qquad \sum_{\mathbf{q} \in \mathbb{N}^n} \rho(\mathbf{q}) = 1$$

Each pattern in the summary S constrains the space for features/words occurring in the pattern:

$$\forall \mathbf{b} \in dom(S) :\ \ S[\mathbf{b}] = \sum_{\mathbf{q} \supseteq \mathbf{b}} \rho(\mathbf{q})^3 \tag{1}$$

The space Ω_S is the set of all query logs, or equivalently the set of all distributions of queries, that obey these constraints. From the observer's perspective, the distribution ρ that the summary conveys is ambiguous: We model this ambiguity with a random variable $\mathcal{P}_S$ with support Ω_S. The actual distribution derived from the log, denoted ρ^*, must appear in Ω_S (i.e., $p(P_S = \rho^*) > 0$). Of the remaining distributions admitted by Ω_S, it is possible that some are more likely than others. For example, an observer might already be aware that field `status` always co-occurs with its table `Messages`. This prior knowledge may be modeled as a prior on the distribution of $\mathcal{P}_S$ or by an additional constraint in Equation 1. Task-specific priors can be defined for special use cases. However, for the purposes of this paper, we take

[2]There are other type of marginals which can be used as carriers of information content. We explain our choice of the specific marginal in Appendix B

[3]The dual constraints $1 - S[\mathbf{b}] = \sum_{\mathbf{q} \not\supseteq \mathbf{b}} \rho(\mathbf{q})$ are omitted as they are redundant under constraint $\sum_{\mathbf{q} \in \mathbb{N}^n} \rho(\mathbf{q}) = 1$

the uninformed prior and assume that $\mathcal{P}_S$ is uniformly distributed over Ω_S:

$$p(\mathcal{P}_S = \rho) = \begin{cases} \frac{1}{|\Omega_S|} & \text{if } \rho \in \Omega_S \\ 0 & \text{otherwise} \end{cases}$$

Naive Summaries. One specific family of summaries that treats each feature as being independent (e.g., as in Figure 1a) is of particular interest to us. Because we will return to it throughout the rest of the paper we give it a specific name: a *naive summary*. A naive summary conveys all patterns with non-zero marginal probabilities and exactly one distinct feature. That is, a naive summary includes each pattern $(0, \ldots, 0, x_i, 0, \ldots, 0)$ for $x_i \in [1, K]$ where K is the lowest integer such that $p(X_i \geq K) = 0$. Equivalently, a naive summary communicates for each word, a categorical distribution:

$$X_i \sim \mathit{Categorical}(K, \Theta_i)$$

where $\Theta_i = (\theta_{i,1}, \ldots, \theta_{i,j}, \ldots, \theta_{i,K})$ is the probability of a query selected uniformly at random from the log having exactly j instances of word i.

4.3 Idealized Representativeness Measures

The representativeness of a summary can be considered from two related, but subtly distinct perspectives: (1) *Ambiguity* measures how much room the summary leaves for interpretation, and (2) *Deviation* measures how reliably the summary approximates the target distribution $p(Q)$.

Ambiguity. The ambiguity of a summary is measured by entropy of the random variable $\mathcal{P}_S$:

$$\mathrm{a}(S) = \sum_{\rho} P(\mathcal{P}_S = \rho) \log \left(P(\mathcal{P}_S = \rho)\right)$$

The higher the entropy, the less precisely S identifies any one specific distribution ρ. Lower entropies indicate a more informative, precise summary.

Deviation. Deviation compares any permitted distribution ρ with the designated true distribution $\rho^* \equiv p(Q)$, measured by the Kullback-Leibler (K-L) divergence[31]:

$$\mathrm{d}(S) \approx \mathcal{D}_{KL}(\rho^* || \rho) = \sum_{\mathbf{q} \in L} \rho^*(\mathbf{q}) \log \frac{\rho^*(\mathbf{q})}{\rho(\mathbf{q})}$$

In principle, we can measure deviation by using the expectation of the K-L divergence over all permitted $\rho \in \Omega_S$:

$$\mathbb{E}\left[\mathcal{D}_{KL}(\rho^* || \mathcal{P}_S)\right] = \sum_{\rho \in \Omega_S} P(\mathcal{P}_S = \rho) \cdot \mathcal{D}_{KL}(\rho^* || \rho)$$

However, there are two limitations to practically compute the two idealized measures. First, K-L divergence is not defined from any probability measure ρ^* that is not *absolutely continuous* with respect to a second (denoted $\rho^* \ll \rho$). In other words, deviation is only defined on variable $\hat{\mathcal{P}}$ with regulated probability measure $\neg(\rho^* \ll \rho) \rightarrow P(\hat{\mathcal{P}} = \rho) = 0$. Second, neither deviation nor ambiguity has a closed-form formula. Thus, we introduce practical computation of representativeness in next section.

5. PRACTICAL REPRESENTATIVENESS

Because there is no closed form representation for the Ambiguity or Deviation of a summary, computing them requires enumerating the full space Ω_S. This is not practical, so an approximation is required. One approach would be to sample from Ω_S (as discussed in Appendix C), but this is still an computationally expensive proposition as we show in our experiments. In this section we propose a faster approach: We identify the maximal entropy distribition $\overline{\rho}$ from Ω_S, and use it to approximate Deviation.

5.1 Summary Error

Inspired by the principle of maximum entropy [22], we select a single distinguished representative distribution $\overline{\rho}_S$ from the space Ω_S:

$$\overline{\rho}_S = \arg\min_{\rho \in \Omega_S} -\mathcal{H}(\rho) \qquad \text{where } \mathcal{H}(\rho) = \sum_{\mathbf{q} \in \mathbb{N}^n} -\rho(\mathbf{q}) \log \rho(\mathbf{q})$$

Maximizing an objective function belonging to the exponential family (entropy in our case) under a mixture of linear equalities/inequalities is a convex optimization problem [4] which guarantees a *unique* solution and can be efficiently solved, for example with *iterative scaling* [10], the cvx toolkit [16, 36], or numerous other approaches.

Specially in the case of naive summaries, we can assume independence across word occurrences X_i. Under this assumption, $\overline{\rho}_S$ has a closed-form solution:

$$\overline{\rho}_S(\mathbf{q}) = \prod_i p(X_i = x_i) \qquad \text{where } \mathbf{q} = (x_1, \ldots, x_n) \qquad (2)$$

Summary Error. Using the representative distribution ρ_S, we define summary error $e(S)$ based on the entropy difference from the space representative to the true distribution:

$$e(S) = \mathcal{H}(\overline{\rho}_S) - \mathcal{H}(\rho^*) \qquad \text{where } \overline{\rho}_S = \arg\min_{\rho \in \Omega_S} -\mathcal{H}(\rho)$$

Relationship to K-L Divergence. Summary error is closely related to the K-L divergence from the representative distribution $\overline{\rho}$ to the true distribution ρ, denoted $\mathcal{D}_{KL}(\rho^* || \overline{\rho}_S)$.

$$\mathcal{D}_{KL}(\rho^* || \overline{\rho}_S) = \mathcal{H}(\rho^*, \overline{\rho}_S) - \mathcal{H}(\rho^*)$$
$$\text{where } \mathcal{H}(\rho^*, \overline{\rho}_S) = \sum_{\mathbf{q}} -\rho^*(\mathbf{q}) \log \overline{\rho}_S(\mathbf{q})$$

$\mathcal{H}(\rho^*, \overline{\rho}_S)$ is called the *cross-entropy*. Replacing cross-entropy by entropy $\mathcal{H}(\overline{\rho}_S)$, the formula becomes the same as summary error. Though cross entropy is different from entropy in general, e.g. it is only defined when $\rho^* \ll \overline{\rho}_S$, they are closely correlated. Specially, under the case of naive summaries, they are equivalent.

LEMMA 1. *For any naive summary S, $\mathcal{H}(\rho^*, \overline{\rho}_S) = \mathcal{H}(\overline{\rho}_S)$*

PROOF. With $\mathbf{q} = (x_1, \ldots, x_n)$ and applying Equation 2:

$$\begin{aligned} \sum_{\mathbf{q} \in L} -\rho^*(\mathbf{q}) \log \overline{\rho}_S(\mathbf{q}) &= -\sum_{\mathbf{q}} P(\mathbf{q}) \cdot \sum_i \log P(X_i = x_i) \\ &= -\sum_{i,k} \log P(X_i = k) \cdot \sum_{\mathbf{q} \mid x_i = k} P(\mathbf{q}) \\ &= -\sum_{i,k} \log P(X_i = k) \cdot P(X_i = k) \\ &= \sum_i \mathcal{H}(X_i) \end{aligned}$$

The variables in a naive summary are independent, so: $\sum_i \mathcal{H}(X_i) = \mathcal{H}(\bar{\rho}_S)$, and we have the lemma. □

While we do not provide a similar proof for more general summaries, we show it experimentally in Section 8.3.

5.2 Summary Error vs Ambiguity

In this section we motivate Summary Error by proving that it correlates with Ambiguity. We supplement the theoretical proofs given in this section, with an empirical analysis relating Summary Error to Deviation in Section 8.3. Our proof is organized as follows: We start by defining a partial order based on *containment* and show that for any pair of summaries for which the partial order is defined, a like relationship is implied for both ambiguity and deviation; We then prove that the total ordering given by summary error is a superset of the partial order.

Recall that each summary S defines a space of possible query distributions Ω_S and *containment* relationship between spaces defines a *partial order* over summaries

$$S_1 \leq_\Omega S_2 \quad \equiv \quad \Omega_{S_1} \subseteq \Omega_{S_2}$$

That is, one summary (S_1) precedes another (S_2) when all distributions admitted by the former summary are also admitted by the latter.

Error Captures Containment. We first prove that the total ordering given by summary error is a superset of the partial ordering $\leq_\Omega$.

PROPOSITION 2. *For any two summaries S_1, S_2 with spaces $\Omega_{S_1}, \Omega_{S_2}$ and maximum entropy distributions $\overline{p}_{S_1}, \overline{p}_{S_2}$ it holds that $S_1 \leq_\Omega S_2 \rightarrow \mathcal{H}(\overline{p}_{S_1}) \leq \mathcal{H}(\overline{p}_{S_2}) \equiv e(S_1) \leq e(S_2)$.*

PROOF. $\Omega_{S_2} \supseteq \Omega_{S_1}$ implies Ω_{S_2} contains $\overline{p}_{S_1}$. Since $\mathcal{H}(\overline{p}_{S_2})$ is the distribution with maximum entropy in space Ω_{S_2}, $\mathcal{H}(\overline{p}_{S_2})$ at least as large as $\mathcal{H}(\overline{p}_{S_1})$. □

Containment Captures Representativeness. Next, we show that partial ordering according to containment implies a like relationship between ambiguity.

PROPOSITION 3. *Assuming $p(\mathcal{P}_s = \rho)$ is uniformly distributed over Ω_S, $S_1 \leq_\Omega S_2 \rightarrow a(S_1) \leq a(S_2)$.*

PROOF. Since $p(\mathcal{P}_s = \rho)$ is uniformly distributed, Equation 4.2 can be applied and $a(S) = \log|\Omega_S|$. Hence $S_1 \leq_\Omega S_2 \rightarrow |\Omega_{S_1}| \leq |\Omega_{S_2}| \rightarrow a(S_1) \leq a(S_2)$ □

5.3 Summary Error and Word-Correlation

Another perspective on summary representativeness is from word-correlation. Consider a pattern $\mathbf{b}$ containing only two distinct words $X_i = x_i, X_j = x_j$, word-correlation captured in the pattern is defined as $\log \frac{p(X_i \geq x_i, X_j \geq x_j)}{p(X_i \geq x_i)p(X_j \geq x_j)}$. In other words, word-correlation measures the log difference between the actual marginal of the pattern and the estimation made by the naive summary which assumes independence among words. The stronger correlation between ith and jth words, with more confidence one can conclude that they co-occur as pattern $\mathbf{b}$ in a query not by chance.

A summary effectively communicates the distribution $p(Q)$, if its patterns capture such word-correlation ignored by naive summaries. For two patterns encoding the same amount of correlation, we prefer the one that occurs more frequently (higher marginal) in queries drawn from the log. As a result, we compute an overall score for ranking patterns by word-correlation:

$$corr\_rank(\mathbf{b}) = p(X_i \geq x_i, X_i \geq x_j) \log \frac{p(X_i \geq x_i, X_j \geq x_j)}{p(X_i \geq x_i)p(X_j \geq x_j)}$$

We show in Section 8.3 that summary error also captures word-correlation: Given the naive summary extended by an additional pattern $\mathbf{b}$, the lower error of the extended summary, the higher $corr\_rank(\mathbf{b})$.

6. PATTERN MIXTURE SUMMARIES

So far, we have explored pattern-based summaries in the context of complete query logs. The query log is treated as a joint distribution of word occurrences, which a summary approximates through patterns. However, patterns provide only positive information—words/features that *do* occur together. Pattern-based summaries begin to break down in settings like logs of mixed workloads, where it is necessary to communicate sets of features that *do not* occur together. Communicating these *anti-correlations* in a consistent way requires adding a significant number of patterns.

In this section, we explore an alternative to trying to capture anti-correlations in a single set of patterns. Instead, we propose a generalization of pattern-based summaries where the log is modeled not as a single probabilistic distribution, but rather as a mixture of several simpler distributions. The resulting summary is likewise a mixture: Each component of the distribution mixture is summarized independently. Hence, we refer to it as a *pattern mixture summary*.

To begin, we will outline a simplified form of pattern mixture summary where we restrict ourselves to mixing purely naive summaries — We refer to this restricted family of summaries as *naive mixture summaries*. Using naive mixture summaries as a motivation, we discuss how to generalize summary error and verbosity to pattern mixture summaries. This in turn helps us to defne a clustering-based approach to constructing naive mixture summaries. Finally, we discuss how this approach may be generalized to construct broader families of summary.

6.1 Example: Naive Mixture Summaries

Consider a toy query log with only 3 conjunctive queries.

1. `SELECT _id FROM Messages WHERE status = 1`
2. `SELECT _id FROM Messages`
3. `SELECT sms_type FROM Messages`

Queries in the log can be encoded using the following words: ⟨ `_id`, `SELECT` ⟩, ⟨ `sms_type`, `SELECT` ⟩, ⟨ `Messages`, `FROM` ⟩, and ⟨ `status = 1`, `WHERE` ⟩. Re-encoding the queries as binary vectors, representing the number of occurrences of each feature, we get the three vectors:

$$\langle\, 1, 0, 1, 1 \,\rangle \qquad \langle\, 1, 0, 1, 0 \,\rangle \qquad \langle\, 0, 1, 1, 0 \,\rangle$$

A naive summary of this log has four independent Bernoulli distributions with parameters ⟨ $p_1 \ldots p_4$ ⟩:

$$\left\langle\, \frac{2}{3}, \frac{1}{3}, 1, \frac{1}{3} \,\right\rangle \equiv$$

SELECT	_id, sms_type
FROM	Messages
WHERE	status = 1

From this summary it is clear that all queries in the log pertain to the Messages table, but that is the only thing we can

determine for certain. For example, this summary obscures the anti-correlation between `_id` and `sms_type`. Similarly, the summary hides the association between `status = 1` and `_id`. Though trivial in this example, relationships like these can be crucial for tasks ranging from database optimization (e.g., to identify useful index structures), to security (e.g., finding unexpected patterns in the workload).

This relationship is captured in the error metric. For example, consider query 1 from the log. The probability of this query is computed as:

$$p(\langle\, 1,0,1,1 \,\rangle) = \quad \frac{2}{3} \cdot \left(1 - \frac{1}{3}\right) \cdot 1 \cdot \frac{1}{3} \quad = \frac{4}{27} \approx 0.148$$

This is a significant difference from true probability of this query (i.e., $\frac{1}{3}$). Conversely non-existant queries like

```
SELECT sms_type FROM Messages WHERE status = 1
```

have non-zero probability. All told, the error value for this summary is $\sum_{i=1,\ldots,4} \mathcal{H}(X_i) - \mathcal{H}(X_1, \ldots, X_4) \approx 0.81$ [4]. To see how we could do better, consider what happens when we partition the log into two segments, denoted here as L_1 and L_2:

Partition 1 (L_1)		**Partition 2** (L_2)
$(1,0,1,1)$	$(1,0,1,0)$	$(0,1,1,0)$

Generating naive summaries for each partition of the log, we only get one non-integral probability: $p(\texttt{status = 1} \mid L_1) = 0.5$. The result is the following two summary visualizations:

SELECT	`_id`
FROM	`Messages`
WHERE	`status = 1`

SELECT	`sms_type`
FROM	`Messages`
WHERE	

Although there are now two summaries, the summaries are not ambiguous. The feature `status = 1` appears in exactly half of the log entries, and is indeed independent of the other features. All other attributes in each summary appear in all queries in their respective entries. Furthermore, the maximum entropy distribution induced by each summary is exactly the distribution of queries in the summarized log and the summary error for each of these two summaries is exactly zero.

6.2 Generalized Representativeness Measures

In this section we generalize *Summary Error* and *Verbosity*, defined under pattern-based summaries, to evaluate pattern mixture summary.

Suppose query log L has been partitioned into K clusters with L_i, S_i, $\overline{\rho}_{S_i}$ and ρ_i^* representing the log of queries, summary, maximum entropy distribution and true distribution for ith cluster respectively. The true distribution ρ^* for the whole log L can be obtained by combining true distribution of each cluster ρ_i^* by

$$\rho^*(\mathbf{q}) = \sum_{i=1,\ldots,K} w_i * \rho_i^*(\mathbf{q}),\ w_i = \frac{|L_i|}{|L|}$$

Generalized Summary Error. Similarly, the maximum entropy distribution $\overline{\rho}_S$ for the whole log can be obtained as

$$\overline{\rho}_S(\mathbf{q}) = \sum_{i=1,\ldots,K} w_i * \overline{\rho}_{S_i}(\mathbf{q})$$

[4]We leave the full error metric computation as an exercise for the reader

The generalized summary error of a pattern mixture summary is then evaluated by

$$\mathcal{H}(\overline{\rho}_S) - \mathcal{H}(\rho^*) = \sum_i w_i * \mathcal{H}(\overline{\rho}_{S_i}) - \sum_i w_i * \mathcal{H}(\rho_i^*) = \sum_i w_i * e(S_i)$$

Generalized Verbosity. Verbosity can be generalized in two ways: (1) *Total verbosity* $\sum_i |S_i|$. It correlates with the number of clusters and measures the verbosity that a user would suffer by examining summaries of all clusters and (2) *Expectation of verbosity* $\sum_k w_k * |S_k|$. It measures the average verbosity weighted by normalized cluster size w_i. To motivate using expectation of verbosity, in addition to total verbosity, we show in Section 7.1.4 that expectation of verbosity *decreases* with increasing number of clusters. This is due to the fact that the less anti-correlated words within clusters, the less expectation of verbosity.

The generalized summary error and verbosity make pattern mixture summaries comparable with pattern-based summaries on equal footing.

7. CONSTRUCTING SUMMARIES

We now discuss our approach to finding log summaries on frontier between generalized summary error and verbosity. Searching the entire space of pattern mixture summaries is infeasible, so we subdivide this problem into two sub-tasks: (1) Finding a naive mixture summary close to the target trade-off point, and (2) Refining this summary into a general pattern mixture summary. As we show in the experiments, naive mixture summaries alone are able to achieve trade-offs that are already close to the frontier.

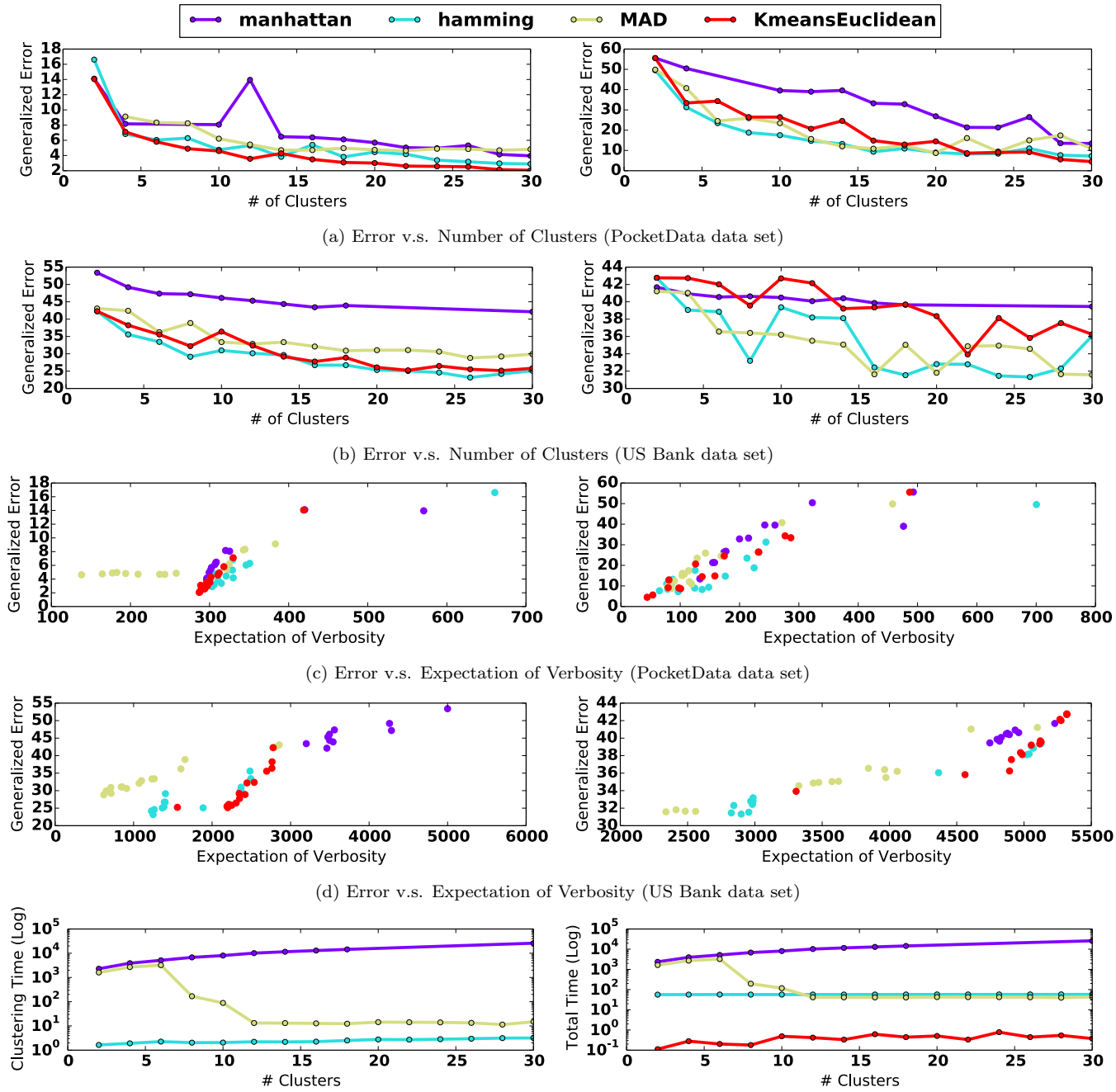
7.1 Constructing Naive Mixture Summaries

In this section, we show how to search for a naive mixture summary. Our approach is based on the observation that each query log, or more precisely each partition of a query log uniquely maps to a naive mixture summary. Thus the problem of searching for a naive mixture summary reduces to the problem of finding log partitions that optimize the tradeoff between generalized summary error and verbosity. We accomplish this by clustering the log and using the resulting clusters as partitions. However for this approach to work, we first need a clustering technique that produces clusters that consistently map to high quality naive mixture summaries (i.e., by finding points close to a desired trade off between summary error and verbosity). To find an appropriate clustering technique, we experimentally evaluated four different forms of clustering: K-Means, spectral clustering with the 2 commonly used[5] distance measures (given in Table 1, and spectral clustering with a novel distance measure that we introduce below called MAD.

Table 1: Distance Measures for Comparison

Distance	Description
Manhattan	$sum(\lvert\mathbf{x} - \mathbf{y}\rvert)$
Hamming	$\frac{Count(\mathbf{x} \neq \mathbf{y})}{Count(\mathbf{x} \neq \mathbf{y}) + Count(\mathbf{x} = \mathbf{y})}$

[5]To simplify illustration, we only pick two representative distances from those we have compared.



(a) Error v.s. Number of Clusters (PocketData data set)

(b) Error v.s. Number of Clusters (US Bank data set)

(c) Error v.s. Expectation of Verbosity (PocketData data set)

(d) Error v.s. Expectation of Verbosity (US Bank data set)

(e) Running Time v.s Number of Clusters (US Bank data set)

Figure 2: Distance Measure Comparison

Experiment Setup. Spectral and KMeans clustering algorithms are implemented by *sklearn* [37] in Python. Data sets and other relevant experiment settings can be found in Section 8.1 and 8.2 respectively. We gradually increase the number of clusters set for these clustering techniques, which can be regarded as continuously sub-clustering the data.

7.1.1 *Data Sub-Clustering*

The comparison based on generalized error (Y-axis) versus the number of clusters (X-axis) is given in Figure 2a and 2b for two data sets respectively. Note that there are two sub-figures in a row for each data set and we first focus on ones on the right.

Trade-off Through Sub-Clustering. A general observation from these figures is that data sub-clustering consistently reduces generalized error of the resulting naive mixture summaries. This shows that we are getting closer to the target trade-off point at each step we sacrifice conciseness of the naive mixture summary by creating more clusters/partitions. In fact, hierarchically sub-clustering the data offers a

SELECT	sms_type, external_ids, _time, _id
FROM	messages
WHERE	(sms_type=1) ∧ (sms_type=0) ∧ (status=1) ∧ (_time≥14260) ∧ (transport_type=3)

(a) *Parent Level*: Naive summary of a mixture of two workloads.

SELECT	sms_type,_time
FROM	messages
WHERE	(sms_type=1)∧ (sms_type=0)∧ (_time≥14260)

SELECT	_id, external_ids
FROM	messages
WHERE	(status=1)∧ (transport_type=3)

(b) *Child Level*: Two branches that separate the mixture.

Figure 3: **Two-level example naive mixture summary**

finer control on trading off between generalized error versus verbosity. Figure 3 shows an example of two-level hierarchy of naive mixture summary. The naive summaries on the child level (Figure 3b) provides a cleaner view (i.e. less verbosity and error for each separate summary) of workload mixture in the query log by separating them. However, the trade-off is that, the total verbosity of two naive summaries on the child level is larger than the one on the parent level. To extend the hierarchy for further reducing generalized error, one can flexibly choose to sub-cluster either left, right or both branches on the child level.

Distance Measures For Sub-Clustering. However, Figure 2a and 2b also show that the speed that sub-clustering approaches the target trade-off point (i.e slope of error reduction with increasing number of clusters) varies under different distance measures and hamming distance is significantly faster than manhattan. To study the cause of such an difference between these two distance measures, we further investigated the data sets and found two limitations of traditional distance measures.

7.1.2 Limitations of Traditional Distance Measures

Word-Correlation-Ignorant. Most distance measures accumulate the similarity/distance contributed by features as if they were independent. Recognizing that features/-words are correlated, and that two queries overlap in one feature will likely overlap in the others is crucial for generating clustering results that map to high quality naive mixture summaries. To illustrate, consider the following example queries.

Example 4. *Word-Correlation*

1. `SELECT _id,...,time_stamp FROM Conversations WHERE status = active`

2. `SELECT _id,...,time_stamp FROM Conversations WHERE status = active AND chat_id = NULL AND blocked = true`

Words in `SELECT` clause from `_id` to `time_stamp` are correlated and shared across two queries in the example. Despite the large number of words shared between queries in above example, the tasks that they are performing are quite different. Specifically, query 1 is scanning all active conversations and query 2 is checking whether active conversations having no chat is blocked. Ignoring word correlation allows similarity dominating dissimilarity between queries, making traditional distance measures fail to correctly distinguish queries apart through sub-clustering. Hamming distance also suffers from this limitation but it is more aggressive in distinguishing queries (i.e. only exact matches on features contribute to similarity).

Query-Multiplicity-Ignorant. The frequency that a query is repeatedly issued refers to query *multiplicity*, which is part of query distribution $p(Q)$. Incorporating word-correlation into distance measures requires taking query multiplicities into consideration. However, for scalability, clustering algorithms are fed with *distinct* queries in the log, making distance measures unaware of query multiplicities.

7.1.3 Multiplicity-Aware Distance

To overcome these limitations, we propose a multiplicity-aware distance (MAD)

$$d_m(\mathbf{q}_1, \mathbf{q}_2) = p(Q \supseteq \mathbf{b}) \log \frac{p(Q \supseteq \mathbf{b})}{\min(p(Q = \mathbf{q}_1 \mid L), p(Q = \mathbf{q}_2 \mid L))}$$

Pattern $\mathbf{b}$ encapsulates the maximal bag of words shared between queries $\mathbf{q}_1, \mathbf{q}_2$, which may be correlated. Left multiplier $p(Q \supseteq \mathbf{b})$ discounts the similarity contributed by shared pattern $\mathbf{b}$, such that the more it is commonly shared by other queries, the less it contributes to query similarity. Note that $\mathbf{b} = \emptyset \rightarrow p(Q \supseteq \mathbf{b}) = 1$. Right multiplier is the log difference from the marginal $p(Q \supseteq \mathbf{b})$ to the minimum of query probabilities $p(Q = \mathbf{q}_i \mid L), i = 1, 2$. The more this minimum probability differ from the marginal of their shared pattern, the larger distance. The resulting distance measure is non-negative, symmetric and reaches zero when $\mathbf{q}_1$ and $\mathbf{q}_2$ are identical.

7.1.4 Selecting a Clustering Technique

In order to select a proper clustering technique for constructing naive mixture summaries, we incorporate MAD distance measure with spectral clustering into comparison and provides a thorough analysis on comparison results shown in Figure 2. Figure 2 is organized as three components: (1) Error v.s. Number Of Clusters, shown in Figure 2a and 2b; (2) Error v.s. Expectation of Verbosity, as shown in Figure 2c and 2d; (3) Running time comparison, as shown in Figure 2e[6]. Note that, except for Figure 2e, all figures consist of two sub-figures in a row with left/right sub-figure ignoring/considering query multiplicities when evaluating generalized error.

Error v.s. Number Of Clusters. We first focus on figures that consider query multiplicities (right sub-figures). Except for manhattan distance with spectral clustering, clustering techniques under comparison perform similarly under PocketData. US bank data set is harder to summarize, with respect to minimum generalized error that all clustering techniques can achieve. MAD works the best under this harder case. By comparing figures that ignore (left) and consider (right) query multiplicities, we observe that query multiplicities greatly influence the generalized error of naive mixture summaries which should not be ignored in general[7].

Error v.s. Expectation of Verbosity. We observe from Figure 2c and 2d that expectation of verbosity positively

[6]We only show running time comparison on US bank data set as the one for PocketData is similar.

[7]Query multiplicities may be ignored in tasks like outlier detection, where infrequent queries are considered equally important as frequent ones.

correlates with expectation of error. In other words, expectation of verbosity is a measure very different from total verbosity. Specially, for PocketData, we observe that MAD with spectral clustering is able to reduce the expectation of verbosity much further than other clustering techniques. This is related to the limitation of word-correlation (See Section 7.1.2). More precisely, the verbosity of a naive summary increases when a large number of words are *anti-correlated* within the cluster. Sub-clustering reduces anti-correlation in each sub-cluster. However, traditional distance measures fail to further distinguish queries into sub-clusters because similarity contributed by correlated words dominates dissimilarity.

Running Time Comparison. A pair distance matrix is required by spectral clustering and should also be considered in running time comparison. Hence we incorporate two sub-figures in a row. Y-axis of the left/right sub-figure represent the time that ignores/considers distance matrix computation time. Note that K-Means does not show up in left sub-figure as it does not require computing distance matrix. One can observe from left sub-figure that running time of spectral clustering is greatly influenced by distance measures. MAD and hamming distance are significantly faster than manhattan. With respect to total running time, K-Means is orders of magnitude faster than spectral clustering.

7.2 Refining Naive Mixture Summaries

To further refine a naive mixture summaries when sub-clustering fail to reach the desired trade-off point, one can selectively choose a set of its clusters where state-of-the art pattern-based summarization techniques are piggybacked. One can then choose either to replace or extend existing naive summaries on selected clusters by additional patterns mined by pattern-based techniques.

Sub-Clustering v.s. Piggybacking Pattern-Based Approaches. Since the goal of pattern-based summarization techniques is to achieve largest information gain by introducing smallest verbosity, piggybacking them is often more *efficient* than sub-clustering with respect to summary error reduction per verbosity increase. However, pattern-based approaches suffer from potentially lower computational efficiency. We empirically show that the time of constructing naive mixture summaries is significantly lower than piggybacking our selected state-of-the-art pattern-based approaches in Section 8.4. In additional, we also compare naive mixture summaries with/without piggybacking pattern-based approaches based on generalized error and verbosity.

8. EXPERIMENTS

In this section, we design experiments to motivate using summary error as one of the summary evaluation measures and constructing pattern mixture summaries as the summarization approach.

8.1 Data Sets

We use two specific query logs in the experiment: (1) SQL query logs of the Google+ app extracted from PocketData dataset [26] over a period of one month and (2) SQL query log that capture all query activity on the majority of databases at a major US bank over a period of approximately 30 hours.

The PocketData-Google+ query log. The query log consists of SQL logs that capture all database activities of 11 Android phones. We selected Google+ application for our study since it is one of the few applications where all users created a workload.

The US bank query log. These logs are anonymized by replacing all constants with hash values generated by SHA-256, and manually vetted for safety. Of the nearly 73 million database operations captured, 58 million are not directly queries, but rather invocations of stored procedures and 13 million not able to be parsed by standard SQL parser. Among the rest of the 2.3 million parsed SQL queries, since we are focusing on conjunctive queries, we base our analysis on the 1.25 million valid `SELECT` queries.

A summary of these two datasets is given in Table 2.

Table 2: Summary of PocketData and US bank query logs

Statistics	PocketData	US bank
# Queries	629582	1244243
# Distinct queries	605	188184
# Distinct queries (w/o const)	605	1712
Max query multiplicity	48651	208742
# Distinct words	863	144708
# Distinct words (w/o const)	863	5290
Average words per query	14.78	16.56

8.2 Common Experiment Settings

Experiments were performed on operating system macOS Sierra with 2.8 GHz Intel Core i7 CPU, 16 GB 1600 MHz DDR3 memory and apple SSD hard drive.

Constant Removal. As one can notice from Table 2, one of the major sources of variation in queries is constants, typically used as parameters in filtering conditions. We ignore constants that generated by SHA-256 hashing.

Query regularization. Not all SQL queries are conjunctive queries. We apply query rewrite rules (similar to [7] to regularize queries into their equivalent form of conjunctive queries.

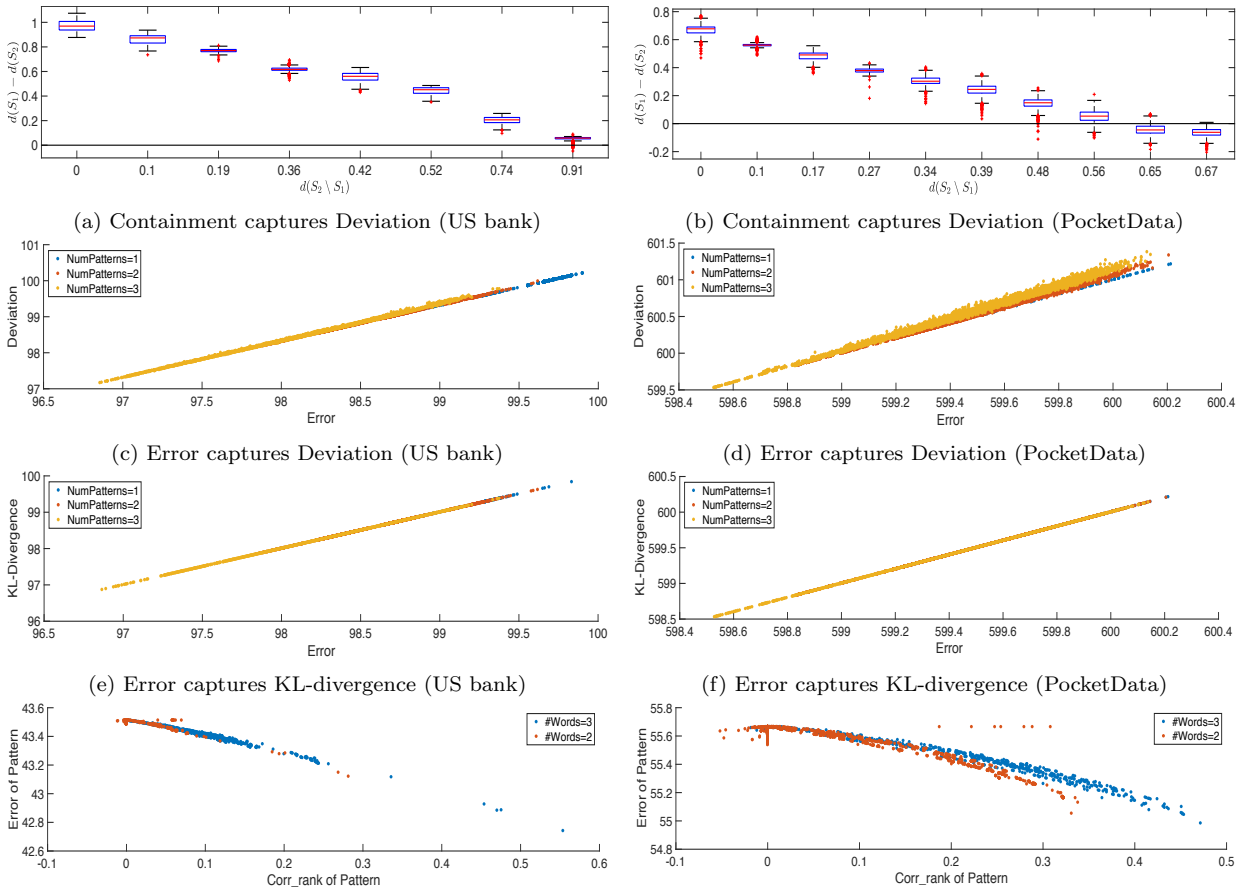
Data Structure. To handle potentially enormous amount of queries in the log, we chose a scalable data structure for storing the multivariate distribution $p(Q)$, which is further described in Appendix D.

Convex Optimization Solving. All convex optimization problems involved in measuring summary error and deviation are solved by *successive approximation heuristic* implemented by Matlab CVX package [16] with Sedumi solver.

8.3 Motivate Summary Error

In this section, we aim at motivating using summary error as the practical alternative for ambiguity and deviation. As it is impractical to enumerate all possible summaries, we choose a subset for both PocketData and US bank data sets. Specifically, we first select a subset of words[8] from which patterns are constructed. We then enumerate combinations of K (K up to 3) patterns as our chosen summaries.

[8]Representing occurrence of ith word as variable X_i, we selected words with $\mathcal{H}(X_i) > T$ where $T = -0.01\log(0.01) - 0.99\log 0.99$.



(a) Containment captures Deviation (US bank)

(b) Containment captures Deviation (PocketData)

(c) Error captures Deviation (US bank)

(d) Error captures Deviation (PocketData)

(e) Error captures KL-divergence (US bank)

(f) Error captures KL-divergence (PocketData)

(g) Error captures Correlation (US bank)

(h) Error captures Correlation (PocketData)

Figure 4: Validating Summary Error Metric

Containment Captures Deviation. We then empirically prove the argument $S_1 \leq_\Omega S_2 \rightarrow d(S_1) \leq d(S_2)$ made in Section 5.2, which completes the chain of reasoning that ordering by summary error is homomorphic to ordering by deviation. Since $S_1 \supseteq S_2 \rightarrow S_1 \leq_\Omega S_2$ and it is much easier to verify $S_1 \supseteq S_2$, we empirically prove the argument $S_1 \supseteq S_2 \rightarrow d(S_1) \leq d(S_2)$. Deviation $d(S)$ is approximated through the two-step sampling described in Appendix C, with 100 samples for the first step and 10000 samples for the second.

The experiment results are shown in Figure 4a and 4b. Given $S_2 \supset S_1$, Y-axis measures difference $d(S_2) - d(S_1)$ in deviation and X-axis measures the deviation of the set-difference $d(S_2 \setminus S_1)$. For better visualization, values in X-axis are normalized by subtracting their minimum value and grouped into 10 bins where box plot is generated. The boxes depict the ranges within standard deviation of Y-values falling into the bins and red marks indicate outlier values. One can observe that the majority of Y-values $d(S_2) - d(S_1)$ stay above zero, which empirically proves $S_2 \supset S_1 \rightarrow d(S_1) < d(S_2)$.

Additive Separability of Deviation. We also observe from Figure 4a and 4b that the larger $d(S_2 \setminus S_1)$, the less $d(S_1) - d(S_2)$. In other words, the following property of deviation holds: $S_2 \supset S_1 \rightarrow d(S_2) - d(S_1) < 0 \ \wedge \ d(S_2 \setminus S_1) \approx d(S_2) - d(S_1)$ where symbol $\approx$ represents positive correlation. This property, mathematically known as *additive separability*, can be interpreted as: The information loss (measured in $d(S_2) - d(S_1)$) by excluding summary $S_2 \setminus S_1$ from S_2 closely correlates with the deviation $d(S_2 \setminus S_1)$.

Error Captures Deviation. Furthermore, we empirically verify that summary error closely correlates with deviation. The experiment results are shown in Figure 4c and 4d. X-axis is summary error $e(S)$ and Y-axis is deviation $d(S)$. One can observe from the figures that the less summary error, the tighter correlation.

Containment Captures Deviation Revisited. From comparing summaries of different verbosity K in Figure 4c and 4d, we observe that $\min_{|S|=K} d(S) < \min_{|S|=K-1} d(S)$, e.g yellow points comparing to red points. Consider $S_2 = \arg\min_{|S|=K} d(S)$ and find the summary $|S_1| = K-1$ where $S_2 \supset S_1$, we have $d(S_2) < d(S_1)$. In other words, this observation is another evidence supporting the argument *containment captures deviation*.

Error Captures KL-Divergence. Next, we empirically prove the argument made in Section 5.1, that summary error is closely correlated with the KL-Divergence from the true distribution ρ^* to space representative distribution $\overline{\rho}_S$. The experiment results that related to this argument are shown in Figure 4e and 4f.

Error Captures Word-Correlation. Last but not least, we show that summary error captures word-correlation, as discussed in Section 5.3. The experiment results related to

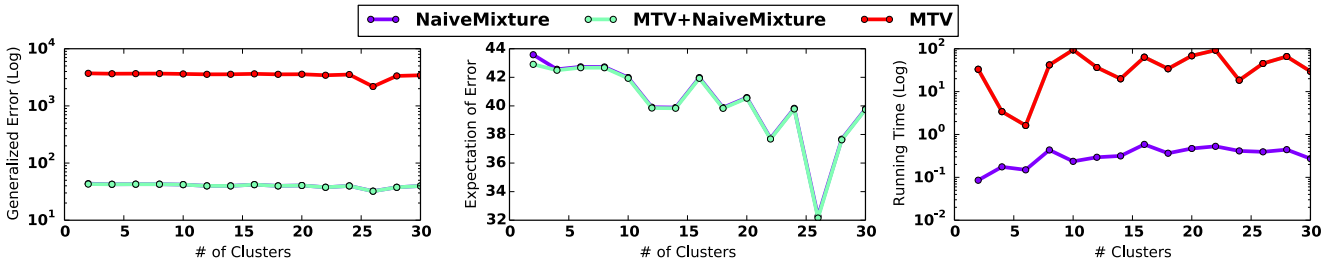


Figure 5: Pattern Mixture Summaries Comparison (US bank query log)

this argument are shown in Figure 4g and 4h. X-axis is word-correlation measure *corr_rank* for specific pattern $\mathbf{b}$ and values in Y-axis is summary error $e(S_{+\mathbf{b}})$ where $S_{+\mathbf{b}}$ denotes the naive summary extended by pattern $\mathbf{b}$. We observe that, the errors of extended summaries almost linearly correlate with *corr_rank*. In addition, *corr_rank* is higher when the pattern $\mathbf{b}$ encodes more words (blue points comparing to red).

8.4 Motivate Pattern Mixture Summaries

After justifying summary error as the summary evaluation measure, we then design experiments to motivate constructing pattern mixture summaries as the preferred summarization technique.

Experiment Setup. Recall in Section 7 that constructing a pattern mixture summary consists of two steps: (1) Constructing a naive mixture summary and then (2) Refining the naive mixture summary by piggybacking on pattern-based summarization techniques. We choose KMeans [21] clustering with Euclidean distance from which naive mixture summaries are constructed. We then choose two state-of-the-art pattern-based algorithms in our experiments: (1) *Laserlight Strategy* [12], which aims at summarizing multi-dimensional data $D = (X_1, \ldots, X_n, A)$ augmented with binary attribute A, or equivalently the conditional probabilities $p(A|X_1, \ldots, X_n)$[9]; (2) *MTV* [33] algorithm, which aims at mining maximally informative patterns as summaries from multi-dimensional data of binary attributes[10]. We set up both algorithms to mine 12 patterns from target clusters[11].

In order to motivate pattern mixture summaries, we construct and compare pattern mixture summaries of three different cases: (1) Only naive mixture summary is constructed; (2) Pattern-based approaches are piggybacked on naive mixture summary and (3) Pattern-based approaches are piggybacked on but existing naive mixture summary is ignored. Case 1 aims at motivating naive mixture summaries and case 2 aims at showing how much improvement on generalized error are contributed by piggybacking pattern-based approaches on naive mixture summaries. Furthermore, since the verbosity of naive summaries on data clusters is potentially much larger than the summaries from pattern-based approaches, one can optionally choose to sacrifice information content for conciseness by summarizing each data cluster solely relying on pattern-based approaches. Hence, case 3 aims at evaluating the information loss by ignoring naive mixture summaries and solely relying on pattern-based approaches. We only show the results for US bank data set as results for PocketData are similar.

8.4.1 Error v.s. Verbosity

Note that there are three sub-figures in a row. The first two sub-figures evaluates generalized error (Y-axis) v.s. the number of clusters (X-axis)[12]. We use two sub-figures for better visualization: The generalized error for case 3 is much higher than case 1 and we need to logarithmically scale the Y-axis (generalized error) for visualizing the comparison on all cases (shown in first sub-figure). As a result, we exclude case 3 and provide in the second sub-figure the comparison results based on an unscaled Y-axis. We observe from the second sub-figure that error reduction brought by piggybacking pattern-based approaches is relatively small for both algorithms. By investigating the data sets, we found that this is because the major source of generalized error for both data sets is *mutual exclusiveness*, a type of anti-correlation that cannot be effectively reduced by pattern-based approaches.

When Pattern-Based Approaches Fail. To illustrate, consider the following example queries in the log.

Example 5. *Mutual Exclusiveness*

```
SELECT _time,_id FROM Messages
WHERE _time>2000

SELECT _time,_id FROM Messages
WHERE _time>3000
```

Queries in the example differ in right hand side (time stamps) of filtering condition `_time` $>$?. These time stamps or more generally the possible parameters of the filtering condition are *mutually exclusive*. Specifically in above example, the number of distinct timestamps for the filtering condition grows with time and none of two can co-occur in the same query. Since each distinct time stamp results in a distinct feature/word, it requires potentially huge number of patterns for encoding mutual exclusiveness among those words and pattern-based approaches are not likely to help. One

[9]Due to implementation restriction of the algorithm, we project the joint distribution of word occurrences $X_1, \ldots, X_n$ onto a limited set of top 100 words ranked by entropy $\mathcal{H}(X_i)$. The binary attribute A is chosen as the word with highest entropy $\mathcal{H}(X_i)$ with binary X_i.

[10]Marginals $p(X_i \geq K)$ where $K > 1$ are thus ignored.

[11]Empirically MTV would take much more running time if this parameter is set higher than 12. In addition, when both algorithms are piggybacked on naive mixture summaries, patterns become more duplicated if the total number of patterns mined is set over 12.

[12]For better visualization, we replace the total verbosity of pattern mixture summaries by the corresponding number of clusters, as they are positively correlated.

can certainly choose to ignore these parameters by taking the risk of information loss. But in general cases where we do not specifically ignore any set of features/words, anti-correlation among words is the bottle neck of both pattern-based and pattern mixture summarization.

Running Time Comparison. We also compare the running time (measured in seconds) of constructing naive mixture summary with the running time of piggybacking pattern-based approaches on all resulting clusters. Note that Y-axis is logarithmically scaled. The experiment result is shown in the third sub-figure. It can be observed that the running time of constructing a naive mixture summary is significantly lower than the chosen pattern mining algorithms, regardless of the number of clusters.

Wrap Up. To wrap up, the take aways are: (1) by sub-clustering, naive mixture summaries are able to provide a precise representation of the query log under controllable conciseness (through hierarchically sub-clustering); (2) To further refine naive mixture summaries, pattern-based summarization approaches can be flexibly piggybacked on naive mixture summaries with evaluable information gain; (3) In cases where feature/word anti-correlation is the major source of summary error, neither sub-clustering nor pattern-based approaches can effectively reduce the summary error.

9. CONCLUSION AND FUTURE WORK

In this paper, we introduced the problem of log summarization and defined two families of summary representations: pattern-based and pattern-mixture. We precisely characterized the information content of a log as a multivariate distribution $p(Q)$ and then offered three principled measures of summary quality: Verbosity, Ambiguity and Deviation. As neither Ambiguity nor Deviation are efficiently computable, we defined a third measure called Summary Error that captures both Ambiguity and Deviation. Thus, we defined log summarization problem as a search over a space of summaries to identify the one that best trades off between Summary Error and Verbosity. Unfortunately, searching for such an ideal summary from the full space of pattern-based summaries is computationally infeasible. We thus restricted our search to a family of pattern mixture summaries which breaks down the distribution $p(Q)$ into several component distributions through clustering. A pattern mixture summary summarizes each component independently and we generalized Summary Error and Verbosity under this setting. To identify an ideal pattern mixture summary, we start by studying its simplified form—naive mixture summaries. We further refined naive mixture summaries by: (1) Sub-clustering the data to scrub through different trade-offs between Verbosity and Summary Error, and (2) Piggybacking on more complex, state-of-the-art pattern-based summarization approaches to create more intricate summaries. We selected two state-of-the-art pattern-based summarization techniques called LaserLight and MTV as comparison methods. On one hand, we experimentally show that naive mixture summaries are faster to create and more informative than comparison methods. On the other hand, we also show that pattern-based approaches are superior with respect to conciseness/verbosity. As a result, we provided guidance on when to piggyback on pattern-based approaches such that a fast-to-create, informative naive mixture summary can also be concise.

We have two directions in mind for our future work: (1) Studying how to concisely convey anti-correlation to further reduce generalized error of pattern mixture summaries; (2) Generalizing bag-of-words data representation to logs of different data types and apply pattern mixture summarization to logs of new data types.

10. REFERENCES

[1] Aligon, J., Golfarelli, M., Marcel, P., Rizzi, S., and Turricchia, E. Similarity measures for olap sessions. *Knowledge and Information Systems 39*, 2 (May 2014), 463–489.

[2] Aouiche, K., Jouve, P.-E., and Darmont, J. Clustering-based materialized view selection in data warehouses. In *ADBIS* (2006).

[3] Barzilay, R., and Elhadad, M. Using lexical chains for text summarization. In *In Proceedings of the ACL Workshop on Intelligent Scalable Text Summarization* (1997), pp. 10–17.

[4] Boyd, S., and Vandenberghe, L. *Convex Optimization*. Cambridge University Press, New York, NY, USA, 2004.

[5] Bruno, N., and Chaudhuri, S. Automatic physical database tuning: A relaxation-based approach. In *ACM SIGMOD* (2005).

[6] Bruno, N., Chaudhuri, S., and Gravano, L. Stholes: A multidimensional workload-aware histogram. In *Proceedings of the 2001 ACM SIGMOD International Conference on Management of Data* (New York, NY, USA, 2001), SIGMOD '01, ACM, pp. 211–222.

[7] Chandra, B., Joseph, M., Radhakrishnan, B., Acharya, S., and Sudarshan, S. Partial marking for automated grading of sql queries. *Proc. VLDB Endow. 9*, 13 (Sept. 2016), 1541–1544.

[8] Chatzopoulou, G., Eirinaki, M., Koshy, S., Mittal, S., Polyzotis, N., and Varman, J. S. V. The QueRIE system for personalized query recommendations. *IEEE Data Eng. Bull.* (2011).

[9] Danaparamita, J., and Gatterbauer, W. Queryviz: Helping users understand sql queries and their patterns. In *EDBT/ICDT* (2011).

[10] Darroch, J. N., and Ratcliff, D. Generalized iterative scaling for log-linear models. *Ann. Math. Statist. 43*, 5 (10 1972), 1470–1480.

[11] Dwork, C. *ICALP 2006, Proceedings, Part II*. Springer, 2006, ch. Differential Privacy.

[12] El Gebaly, K., Agrawal, P., Golab, L., Korn, F., and Srivastava, D. Interpretable and informative explanations of outcomes. *Proc. VLDB Endow. 8*, 1 (Sept. 2014), 61–72.

[13] Erkan, G., and Radev, D. R. Lexrank: Graph-based lexical centrality as salience in text summarization. *J. Artif. Int. Res. 22*, 1 (Dec. 2004), 457–479.

[14] Giacometti, A., Marcel, P., Negre, E., and Soulet, A. Query recommendations for OLAP discovery driven analysis. In *ACM DOLAP* (2009).

[15] Gong, Y., and Liu, X. Generic text summarization using relevance measure and latent semantic analysis. In *Proceedings of the 24th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval* (New York, NY, USA, 2001), SIGIR '01, ACM, pp. 19–25.

[16] Grant, M., and Boyd, S. CVX: Matlab software for disciplined convex programming, version 2.1. `http://cvxr.com/cvx`, Mar. 2014.

[17] Han, J., Cheng, H., Xin, D., and Yan, X. Frequent pattern mining: Current status and future directions. *Data Min. Knowl. Discov. 15*, 1 (Aug. 2007), 55–86.

[18] Han, J., Pei, J., and Yin, Y. Mining frequent patterns without candidate generation. *SIGMOD Rec. 29*, 2 (May 2000), 1–12.

[19] Hovy, E., yew Lin, C., Zhou, L., and Fukumoto, J. Automated summarization evaluation with basic elements. In *In Proceedings of the Fifth Conference on Language Resources and Evaluation (LREC)* (2006).

[20] Hu, L., Ross, K. A., Chang, Y.-C., Lang, C. A., and Zhang, D. Queryscope: visualizing queries for repeatable database tuning. *pVLDB* (2008).

[21] Jain, A. K. Data clustering: 50 years beyond k-means. *Pattern Recogn. Lett. 31*, 8 (June 2010), 651–666.

[22] Jaynes, E., and Bretthorst, G. *Probability Theory: The Logic of Science*. Cambridge University Press, 2003.

[23] Jing, H., Barzilay, R., McKeown, K., and Elhadad, M. Summarization evaluation methods: Experiments and analysis. In *IN AAAI SYMPOSIUM ON INTELLIGENT SUMMARIZATION* (1998), pp. 60–68.

[24] Kamra, A., Terzi, E., and Bertino, E. Detecting anomalous access patterns in relational databases. *VLDBJ* (2007).

[25] Kennedy, O., Ajay, J., Challen, G., and Ziarek, L. Pocket Data: The need for TPC-MOBILE. In *TPC-TC* (2015).

[26] Kennedy, O., Ajay, J. A., Challen, G., and Ziarek, L. Pocket data: The need for tpc-mobile. In *TPC-TC* (2015).

[27] Khoussainova, N., Kwon, Y., Balazinska, M., and Suciu, D. SnipSuggest: context-aware autocompletion for SQL. *pVLDB* (2010).

[28] Kokkalis, A., Vagenas, P., Zervakis, A., Simitsis, A., Koutrika, G., and Ioannidis, Y. Logos: A system for translating queries into narratives. In *ACM SIGMOD* (2012).

[29] Kul, G., Luong, D., Xie, T., Coonan, P., Chandola, V., Kennedy, O., and Upadhyaya, S. Ettu: Analyzing query intents in corporate databases. In *WWW Companion* (2016).

[30] Kul, G., Luong, D., Xie, T., Coonan, P., Chandola, V., Kennedy, O., and Upadhyaya, S. Ettu: Analyzing query intents in corporate databases. In *Proceedings of the 25th International Conference Companion on World Wide Web* (Republic and Canton of Geneva, Switzerland, 2016), WWW '16 Companion, International World Wide Web Conferences Steering Committee, pp. 463–466.

[31] Kullback, S., and Leibler, R. A. On information and sufficiency. *Ann. Math. Statist. 22*, 1 (03 1951), 79–86.

[32] Makiyama, V. H., Raddick, M. J., and Santos, R. D. Text mining applied to SQL queries: A case study for the SDSS SkyServer. In *SIMBig* (2015).

[33] Mampaey, M., Vreeken, J., and Tatti, N. Summarizing data succinctly with the most informative itemsets. *ACM Trans. Knowl. Discov. Data 6*, 4 (Dec. 2012), 16:1–16:42.

[34] Mani, I., House, D., Klein, G., Hirschman, L., Firmin, T., and Sundheim, B. The tipster summac text summarization evaluation, 1999.

[35] Nenkova, A., Passonneau, R., and McKeown, K. The pyramid method: Incorporating human content selection variation in summarization evaluation. *ACM Trans. Speech Lang. Process. 4*, 2 (May 2007).

[36] O'Donoghue, B., Chu, E., Parikh, N., and Boyd, S. Conic optimization via operator splitting and homogeneous self-dual embedding. *Journal of Optimization Theory and Applications 169*, 3 (Jun 2016), 1042–1068.

[37] Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., and Duchesnay, E. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research 12* (2011), 2825–2830.

[38] Radev, D. R., Jing, H., and Budzikowska, M. Centroid-based summarization of multiple documents: Sentence extraction, utility-based evaluation, and user studies. In *Proceedings of the 2000 NAACL-ANLPWorkshop on Automatic Summarization - Volume 4* (Stroudsburg, PA, USA, 2000), NAACL-ANLP-AutoSum '00, Association for Computational Linguistics, pp. 21–30.

[39] Radev, D. R., and Tam, D. Summarization evaluation using relative utility. In *Proceedings of the Twelfth International Conference on Information and Knowledge Management* (New York, NY, USA, 2003), CIKM '03, ACM, pp. 508–511.

[40] Yang, X., Procopiuc, C. M., and Srivastava, D. Recommending join queries via query log analysis. In *IEEE ICDE* (2009).

[41] yew Lin, C. Rouge: a package for automatic evaluation of summaries. pp. 25–26.

APPENDIX

A. NOMENCLATURE

Symbol	Meaning
f	Feature
w	Word
$\mathbf{b}$	Bag of words/Pattern
$\mathbf{b}' \subseteq \mathbf{b}$	$\mathbf{b}'$ is contained in $\mathbf{b}$
$\mathbf{q}$	Query
L	Log, a bag of queries
Q	Query uniformly drawn from L
$p(Q = \mathbf{q} \mid L)$	Probability of $\mathbf{q}$ uniformly drawn from L
$p(Q)$	$p(Q = \mathbf{q} \mid L)$
$p(Q \supseteq \mathbf{b})$	Marginal probability of Q containing $\mathbf{b}$
$corr\_rank(\mathbf{b})$	Word-correlation in pattern $\mathbf{b}$
S_{max}	Full mapping from patterns to marginals
S	Summary, partial mapping $S \subseteq S_{max}$
$dom(\cdot)$	Domain of mapping
$\overline{S}$	Lossless summary
ρ	Distribution of vectorized queries $\mathbf{q} \in \mathbb{N}^n$
Ω_S	Space of ρ constrained by S
$S \leq_\Omega S'$	Partial order over S, $\Omega_S \subseteq \Omega_{S'}$
$\mathcal{P}_S$	ρ randomly drawn from Ω_S
ρ^*	Same as $p(Q)$, $\rho^* \in \Omega_S$
$\mathcal{H}(\cdot)$	Distribution Entropy
$\overline{\rho}_S$	$\arg\max_{\rho \in \Omega_S} \mathcal{H}(\rho)$
$d_m(\mathbf{q}, \mathbf{q}')$	Multiplicity-Aware Distance (MAD)
$d(S)$	Deviation
$a(S)$	Ambiguity
$e(S)$	Summary Error
$\|S\|$	$\|dom(S)\|$, summary Verbosity
$\rho \ll \rho'$	ρ is absolutely continuous w.r.t ρ'
$\mathcal{D}_{KL}(\rho \| \rho')$	K-L Divergence, $\sum_{\mathbf{q} \in \mathbb{N}^n} \rho(\mathbf{q}) \log \frac{\rho(\mathbf{q})}{\rho'(\mathbf{q})}$.
$\mathbb{E}[\cdot]$	Expectation

B. MARGINAL SELECTION

As discussed in Section 4.2, a pattern conveys information content of the log by constraining an arbitrary distribution ρ (zero information content of the log) such that ρ has the same marginal $p(Q \supseteq \mathbf{b})$ or equivalently $p(X_1 \geq x_1, \ldots, X_n \geq x_n)$ where $\mathbf{b} = (x_1, \ldots, x_n)$. We keep including patterns in the summary such that the constrained ρ eventually converges to $p(Q)$.

However, marginal $p(Q \supseteq \mathbf{b})$ is not the only choice. In fact, the marginal probability that a query $\mathbf{q}$ uniformly drawn from the log *exactly matches* (denoted as $\mathbf{q} \sqsupseteq \mathbf{b}$) the pattern $\mathbf{b}$ can be an alternative. Specifically, given $\mathbf{q} = (x_1, \ldots, x_n)$ and $\mathbf{b} = (x'_1, \ldots, x'_n)$,

$$\mathbf{q} \sqsupseteq \mathbf{b} \equiv \forall x'_i > 0,\ x_i = x'_i$$

Given that word occurrences $x_1, \ldots, x_m,\ m \leq n$ are positive in pattern $\mathbf{b}$, the resulting marginal $p(Q \sqsupseteq \mathbf{b})$ is equivalent to $p(X_1 = x_1, \ldots, X_m = x_m)$. Here we explain the rationale on choosing $p(Q \supseteq \mathbf{b})$ over $p(\sqsupseteq \mathbf{b})$ in the following example.

EXAMPLE 6. *Convey the information that, i, jth words never co-occur twice in any query drawn from the log.*

1. *Marginal Type $p(Q \supseteq \mathbf{b})$:* $p(X_i \geq 2, X_j \geq 2) = 0$
2. *Marginal Type $p(Q \sqsupseteq \mathbf{b})$:* $p(X_i = 2, X_j = 2) = 0$,
   $p(X_i = 2 + 1, X_j = 2) = 0$,
   $p(X_i = 2, X_j = 2 + 1) = 0$,
   $\ldots$

In other words, marginal $p(Q \supseteq \mathbf{b})$ is able to convey the objective information using only one pattern. As a conclusion, in general, marginal $p(Q \supseteq \mathbf{b})$ offers more descriptive power than $p(Q \sqsupseteq \mathbf{b})$.

C. SAMPLING FROM SPACE OF DISTRIBUTIONS

Here we discuss sampling from a space of probability distributions.

C.1 Sampling without Constraints

To sample a random distribution $\rho : \mathbb{N}^n \to [0, 1]$ without any constraint, the naive way is to treat ρ as a multi-dimensional random vector $(\rho(\mathbf{q}_1), \ldots, \rho(\mathbf{q}_{|\mathbb{N}^n|}))$ which sums up to 1. However, $|\mathbb{N}^n|$ is too large and we reduce the number of dimensions by grouping queries $\mathbf{q}_i \in \mathbb{N}^n$ into equivalence classes.

Algorithm 1 Sampling

```
1: procedure TwoStepSampling
2:     Step 1:
3:     for each v ∈ 𝔹^m ∧ C_v ≠ ∅ do
4:         V ← V ⋃ v
5:     end for
6:     class_p ← UniRandDistribProb(V,1)
7:     Step 2:
8:     for each v ∈ V do
9:         ρ ← ρ ⋃ UniRandDistribProb(C_v, class_p(v))
10:    end for
11:    return ρ
12: end procedure
13:
14: procedure UniRandDistribProb(Set S, double prob)
15:    for each element e ∈ S do
16:        p(e) ← UniformRandNum(range = [0, 1])
17:    end for
18:    for each element e ∈ S do
19:        p(e) ← prob × p(e) ÷ Σ_e p(e)
20:    end for
21:    return p
22: end procedure
```

Summary-equivalent Classes. The basic idea for grouping is based on containment relationship between query and patterns $\mathbf{b} \in S$ in the summary S. More precisely, if $\mathbf{q}_i \supseteq \mathbf{b}$, it indicates that probability $\rho(\mathbf{q}_i)$ is constrained by marginal $S[\mathbf{b}]$ (see Equation 1 in Section 4.2). As a result, if queries $\mathbf{q}_i, \mathbf{q}_j$ share the same containment relationship with pattern $\mathbf{b}$, probabilities $\rho(\mathbf{q}_i), \rho(\mathbf{q}_j)$ make no difference for satisfying the constraint derived from $\mathbf{b}$. We thus define *pattern-equivalence* as

$$\mathbf{q}_i \equiv_{\mathbf{b}} \mathbf{q}_j \Leftrightarrow I(\mathbf{q}_i, \mathbf{b}) = I(\mathbf{q}_j, \mathbf{b})$$

$I(\mathbf{q}_i, \mathbf{b})$ is the binary indicator function satisfying $I(\mathbf{q}_i, \mathbf{b}) = 1 \equiv \mathbf{q}_i \supseteq \mathbf{b}$. Queries are *summary-equivalent* $\mathbf{q}_i \equiv_S \mathbf{q}_j$ if they are pattern-equivalent for all patterns in the summary.

Ordering patterns in the summary as $\mathbf{b}_1, \ldots, \mathbf{b}_m$, any binary vector $\mathbf{v} \in \mathbb{B}^m$ maps to an equivalence class $\mathcal{C}_\mathbf{v} = \{ \mathbf{q} \mid (I(\mathbf{q}, \mathbf{b}_1), \ldots, I(\mathbf{q}, \mathbf{b}_m)) = \mathbf{v} \wedge \mathbf{q} \in \mathbb{N}^n \}$. Though the number of non-empty equivalent classes may grow as large as $\mathcal{O}(2^m)$, it is much smaller than $|\mathbb{N}^n|$ and creating a random distribution can be divided into two steps as shown in line 1 of algorithm 1.

C.2 Sampling under Constraints

So far we are creating random samples from an unconstrained space of distributions. To make sure ρ produced by the two-step sampling fall within space Ω_S, distributions $class\_p$ produced by Step 1 must obey the linear equality constraints derived from the summary S. Denote the space of all possible distributions $class\_p$ as U and the subspace allowed by the summary as $U_S \subseteq U$. One naive solution is to reject $class\_p \notin U_S$. However, the subspace U_S constrained under linear equality constraints is equivalent to an intersection of *hyperplanes* in the full space U. The volume of U_S is thus so small comparing to that of U, such that any random sample $class\_p \in U$ will *almost never* fall within U_S. To make sampling feasible, we do not reject a sample $class\_p \in U$ but *project* it onto U_S by finding its *closest* counterpart $class\_p' \in U_S$. In other words, given summary S, we assume we are sampling $class\_p'$ from allowed subspace U_S given the probability measure,

$$p(class\_p') \propto \left| \{ \ class\_p \mid class\_p \in U \ \wedge \ proj(class\_p) = class\_p' \ \} \right|$$

The projection can be achieved by solving an optimization problem that minimizes some distance function $d(class\_p, class\_p')$, e.g. $d(class\_p, class\_p') = ||class\_p' - class\_p||$ under the constraints.

D. DATA STRUCTURE

We choose FP Tree [18] as our solution for storing distribution $p(Q)$ because it is lossless and reasonably scalable on large data sets (See *frequent pattern mining* [17]). Here we show that it can seamlessly work for our case by translating any *bag* of words into a *set* of items: Suppose a word w occurs k times in query $\mathbf{q}$. This implies that it also occurs 1 to $k-1$ times. We thus create k correlated items $(w, 1), (w, 2)..(w, k)$ for query $\mathbf{q}$.