Selectivity Estimation on Streaming Spatio-Textual Data Using Local Correlations

†Xiaoyang Wang, ‡Ying Zhang, †Wenjie Zhang, †Xuemin Lin*, †Wei Wang

†*The University Of New South Wales* ‡*The University of Technology, Sydney*

xiaoyangw@cse.unsw.edu.au Ying.Zhang@uts.edu.au {zhangw, lxue, weiw}@cse.unsw.edu.au

ABSTRACT

In this paper, we investigate the selectivity estimation problem for streaming spatio-textual data, which arises in many social network and geo-location applications. Specifically, given a set of continuously and rapidly arriving spatio-textual objects, each of which is described by a geo-location and a short text, we aim to accurately estimate the cardinality of a spatial keyword query on objects seen so far, where a spatial keyword query consists of a search region and a set of query keywords.

To the best of our knowledge, this is the first work to address this important problem. We first extend two existing techniques to solve this problem, and show their limitations. Inspired by two key observations on the "locality" of the correlations among query keywords, we propose a *local correlation* based method by utilizing an augmented adaptive space partition tree (A^2SP-tree for short) to approximately learn a *local Bayesian network* on-the-fly for a given query and estimate its selectivity. A novel *local boosting* approach is presented to further enhance the learning accuracy of local Bayesian networks. Our comprehensive experiments on real-life datasets demonstrate the superior performance of the local correlation based algorithm in terms of estimation accuracy compared to other competitors.

1. INTRODUCTION

With the advances in geo-positioning technologies and geo-location services, there is a rapidly growing amount of spatio-textual objects collected in many applications such as location based services and social networks, in which an object is described by its spatial location and a set of terms (keywords). For instance, in the GPS navigation system, a POI (point of interest) is a geographically anchored pushpin that someone may find useful or interesting, which is usually annotated with textual information (e.g., descriptions and users' reviews). In social media (e.g., Flickr, Facebook, FourSquare, Twitter and Weibo), a large number of posts and photos are usually associated with a geo-position as well as a short text. In the above applications, a large volume of spatio-textual objects may continuously arrive with high speed. For instance, recently it is reported[1] that there are about 30 millions people sending geo-tagged data out into the Twitterverse, and 2.2 percent of tweets (about 4.4 million tweets a day) provide location data together with the text of their posts. Consequently, there is an emerging call for efficient data analytics techniques to make sense of the streaming spatio-textual objects. In this paper, we investigate the problem of selectivity estimation on streaming spatio-textual objects which counts the number of spatio-textual objects for given spatial and textual constraints. More specifically, assuming spatio-textual objects continuously arrive, for a given search region (e.g., rectangle or circle) and a set of query keywords, we aim to find the number of objects seen so far, each of which falls in the search region and contains all query keywords. Selectivity estimation is an essential and fundamental tool for streaming data analytics with a wide spectrum of applications, because the selectivity estimation can provide users with fast, useful feedback so that they can identify the truly interesting region and then focus their exploration quickly and effectively.

Figure 1: Motivation Example

Below are motivation examples of selectivity estimation for streaming spatio-textual objects.

Example 1 (Motivation). *Twitter has brought great advertising opportunities to aid advertisers to approach the promising users at the right moment and in the right context [12]. Suppose that the points in Figure 1 represent a set of recent tweets, each of which is described by the text*

*Corresponding author

Proceedings of the VLDB Endowment, Vol. 8, No. 2

[1] http://www.futurity.org/tweets-give-info-location

of the post and the sender's location. Then a sales manager may decide if it is cost-effective to advertise their products in a particular region by SMS broadcast service or on-site promotion. For instance, it will be a positive sign to setup an on-campus promotion for google glass if many posts in the University contain both keywords "glass" and "google". Similarly, a social researcher may find out the degree of the concerns of some social events in a particular region. For example, she may count the number of posts containing both keywords "batman" and "leukemia" for the Batkid event at San Francisco where a 5-year-old boy with leukemia got wish to be Batman for a day. Similar example goes to the keywords "election" and "Obama" for the US election.

Challenges. The main challenges of the selectivity estimation on streaming spatio-textual objects lie in the following three aspects. *Firstly*, the massive spatio-textual objects continuously arrive at a rapid rate in many applications. Given a memory budget, we need to effectively build a concise summary to support selectivity estimation with high accuracy and low query latency. *Secondly*, it is challenging for the summary to support both spatial and keyword constraints, as existing methods can only handle one type of the constraints. *Thirdly*, query keywords typically have correlations among them in complex manners, yet it is unrealistic to capture the correlations among all keywords appearing in the data due to the sheer amount of distinct number of keywords.

As a result, to the best of our knowledge, none of the traditional selectivity estimation techniques is applicable to the problem studied in the paper, as they need off-line computation of the summaries, or they cannot efficiently support selectivity estimation on data with a large number of attributes (i.e., keywords) as well as geo-locations. As shown in Section 3, two categories of techniques can be extended to support the selectivity estimation problem are spatial stream range counting (**RC** for short) estimators (e.g., [14, 9]) and distinct values (**DVs** for short) estimators (e.g., [3, 10, 18]). However, according to our theoretical analysis and empirical study, both approaches have their inherent limitations. Assuming a RC estimator is constructed for each individual query keyword, we can come up with the selectivity estimation algorithm, denoted by **RC-E**, with independence assumption of the query keywords. However, our empirical study shows that the accuracy of this approach is unsatisfactory even correlations among query keywords are available. On the other hand, DVs estimator based techniques for multi-set operations can be employed, since the selectivity estimation of spatial keyword query can be regarded as the problem of estimating multi-set intersection where each query keyword is associated with a set of objects within the search region. Nevertheless, the algorithm directly applying DVs estimator, denoted by **DVs-E**, suffers from two drawbacks. Firstly, DVs estimators need a fairly large number of samples to achieve a decent accuracy. Unfortunately, in our study the number of samples which survives both spatial and keyword constraints may be rather small, and hence leads to unreliable estimates. Secondly, when there is only one query keyword, the performance of DVs-E is much poorer than that of RC-E. This is not surprising because RC estimators are dedicated summaries to support selectivity estimation for range search.

Motivated by the limitations of the above methods, we aim to develop a novel technique for the problem of selectivity estimation on streaming spatio-textual objects. In a nutshell, we exploit the "locality" of probabilistic influence among the keywords of the streaming spatio-textual objects and derive the cardinality of the spatial keyword query based on **local correlations** among query keywords, i.e., correlations obtained from objects within the search region. Regarding local correlations, we have two key observations in this paper. Given a search region, the local correlation of two keywords may arbitrarily deviate from their corresponding *global correlation*, i.e., the correlations of terms regarding all objects. For example, "batman" and "leukemia" may have little global correlation, but exhibit strong local correlation in San Francisco. Consequently, we learn the local correlations of the query keywords on-the-fly based on the objects within the search region, which is captured by the popular Bayesian networks model in this paper. The other important observation is that local correlations are typically well preserved if we enlarge the search region appropriately. For example, the local correlation between the two identical keywords will be still strong in California, or even USA. This leads to our novel *local boosting* approach which significantly improves the learning accuracy of the local correlations as it overcomes the data sparsity issue. To effectively support the local correlation based selectivity estimation, we propose an augmented adaptive space partition tree, namely **A^{2}SP-tree**, by integrating the strengths of RC estimator and DVs estimator.

Contributions. Our principle contributions of this paper are summarized as follows.

- This is the first work to systematically study the problem of selectivity estimation for streaming spatio-textual objects against spatial keyword queries, which is an essential tool for streaming data analytics in a wide spectrum of applications.
- We develop two baseline algorithms, namely RC-E and DVs-E, by extending RC estimator and DVs estimator techniques, respectively. Key observations and insights about the limitations of two approaches are reported through theoretical analysis and empirical study.
- We advocate a new computing paradigm using local correlations among the query keywords. An augmented adaptive space partition tree structure (A^2SP-tree for short), is introduced to incorporate the strengths of RC estimator and DVs estimator techniques. We develop local correlation based algorithm, namely **BN-E**, to learn local correlations among query keywords on-the-fly based on a *local Bayesian network* and derive the selectivity estimation. A novel *local boosting* approach is proposed to enhance the estimate accuracy.
- We extend our techniques to support sliding window model and boolean spatial keyword query.
- Comprehensive experiments on real-life datasets demonstrate superior performance of the proposed local correlation based algorithm.

Organization of the paper. The rest of the paper is organized as follows. Section 2 gives the problem definition, briefly surveys the most related work and necessary techniques. Section 3 proposes our first two algorithms by extending existing techniques. Section 4 presents our third algorithm which captures local correlation effectively and efficiently. Several extensions are discussed in Section 5. The experimental results are reported and analyzed in Section 6. We conclude the paper in Section 7.

2. BACKGROUND

In this section, we first formally define the problem of selectivity estimation on steaming spatio-textual data. Then we introduce the related existing work as well as the key techniques employed in the paper. Table 1 summarizes the notations frequently used throughout the paper.

Notation	Meaning
o, p	spatio-textual object
$\mathcal{D}$	a stream of spatio-textual objects
$\mathcal{V}$	the vocabulary of terms (keywords)
J	domain of 2-dimensional space $[0, J]^2$
q	a spatial keyword query
$q.R$	query region of q
t, t_i, t_j	a term (keyword)
$\mathcal{D}(t)$	objects in $\mathcal{D}$ contain term t
$o.\mathcal{V}$ ($q.\mathcal{V}$)	terms (keywords) of o (q)
n	the number of objects in $\mathcal{D}$
n_i	the number of objects with term t_i
m	the number of query keywords
A ($\hat{A}$)	selectivity of q (estimation of A)
$\mathcal{L}$ ($\mathcal{L}_i$)	KMV synopses (KMV synopses w.r.t t_i)
$\mathcal{T}$ ($\mathcal{T}_i$)	ASP-tree (ASP-tree w.r.t t_i)
$\mathcal{B}$	a Bayesian network
T_i, X_i	binary random variable w.r.t term t_i
$\mathcal{X}$	a set of random variables
$P_{\mathcal{B}}(X_i\|Pa(X_i))$	CPD of a random variable X_i in $\mathcal{B}$
$P_{\mathcal{B}}(\mathcal{X}=1)$	the probability that $\mathcal{X}$ equals 1 in $\mathcal{B}$

Table 1: The Summary of Notations

2.1 Problem Definition

In this paper, a spatio-textual object o is described by a spatial point in a 2-dimensional space $[0, J]^2$ and a set of terms (aka., keywords) from the vocabulary $\mathcal{V}$, denoted by $o.loc$ and $o.\mathcal{V}$, respectively. A spatial keyword query q, consisting of a region $q.R$ and a set of query keywords $q.\mathcal{V}$, retrieves all objects that fall in the region $q.R$ and contain **all** the query keywords, i.e., $o.loc \in q.R$ and $o.\mathcal{V} \supseteq q.\mathcal{V}$. Note that we discuss how to extend our techniques to support arbitrary boolean spatial keyword query in Section 5, in which conjunctions (ANDs), disjunctions (ORs) and negations (NOTs) are considered. In this paper, we aim to maintain a summary of the streaming spatio-textual objects to estimate the selectivity of spatial keyword queries. Following is the formal definition of the problem.

Problem Statement. We investigate the problem of continuously maintaining a summary $\mathcal{S}$ within a space budget of B over a stream of spatio-textual objects $\mathcal{D}$ such that, at any time, $\mathcal{S}$ can be used to approximate the selectivity of the spatial keyword queries. The aim is to achieve high estimation accuracy under the given memory budget.

In this paper hereafter, whenever there is no ambiguity, "spatio-textual object" is abbreviated to "object" and $\mathcal{D}$ represents the streaming objects seen so far when a query q is issued. We use A to denote the cardinality of the query q while $\hat{A}$ is an estimation of A. n and m represent the number of objects in $\mathcal{D}$ and the number of query keywords, respectively. We might use "term" and "keyword" interchangeably for better understanding of the paper.

2.2 Related Work

To the best of our knowledge, there is no existing work on the selectivity estimation of spatial keyword search on streaming spatio-textual objects. In this subsection, we review two important categories of works that are closely related to the problem studied in this paper.

2.2.1 Searching and Mining Spatio-Textual Data

The study of spatial keyword search and mining has attracted a great attention from the commercial organizations and research communities due to an ever-increasing amount of spatio-textual objects in many emerging applications. One of the most representative queries is the *spatial keyword search* which aims to find a set of spatio-textual objects based on the spatial proximity and query keywords constraints (e.g., [24, 6]). There are many important variants in the literature with different focuses such as the spatial keyword ranking query (e.g., [25]) and collective spatial keyword search (e.g., [17]). Recently, many works have been dedicated to make sense of streaming spatio-textual objects, especially the microblogs with geo-locations, including localized event detection [1], twitter advertising [12], geo-correlated information trends detection [4], frequent spatio-temporal term queries [21], spatio-textual object filtering [5, 16], etc. However, none of the existing work investigates the problem of selectivity estimation on streaming spatio-textual data.

2.2.2 Selectivity Estimation

The problem of selectivity estimation has been extensively studied for a large variety of queries such as range queries (e.g., [20, 15, 22, 14, 9]), boolean queries (e.g., [7]), relational join (e.g., [23]), spatial join (e.g.,[11]) and set intersection (e.g., [18]). Nevertheless, many of the techniques developed in the above work need off-line computation of the data summaries or are sensitive to the dimensionality of the data, and hence cannot be adopted to the problem of selectivity estimation for streaming spatio-textual objects because of the rapid arriving speed of massive objects and the large number of distinct terms. Moreover, most of the existing works do not consider the locations of the objects.

Nevertheless, as shown in Section 3, two categories of techniques can be naturally extended to tackle the problem studied in this paper. The existing techniques on range counting for steaming spatial data (e.g., [14, 9]) can approximate the number of objects within a search region, and hence are extended to support selectivity estimation of spatial keyword search with independence assumption in Section 3.1. Since DVs estimators (e.g., KMV [3], bottom-k [10, 18], and min-hash [18]) can effectively support size estimation for multi-set operations (e.g., union and intersection), they are widely used for the problems of size estimation under different contexts (e.g.,[7, 13, 27]). In Section 3.2, we also show how to extend DVs estimator technique to the problem studied in this paper.

2.3 Preliminaries

In this subsection, we introduce three important techniques employed in this paper.

2.3.1 Adaptive Space Partition (ASP) Tree

In [14], Hershberger *et al.* propose the adaptive spatial partition tree (***ASP*-tree**), denoted by $\mathcal{T}$, assuming the points of the stream $\mathcal{D}$ are located in a 2-dimensional $[0, J]^2$. As shown in Figure 2, the ASP-tree is a 4-ary compressed version of the standard quadtree tree with maximal height $\log(J)$. Each node $e \in \mathcal{T}$ is associated with a cell and a counter, denoted by $e.c$ and $e.v$, respectively. Each point will be counted by exactly one node, i.e., $\sum_{e\in\mathcal{T}} e.v = n$. A

split threshold α ($0 \leq \alpha \leq 1$) is employed to control the size of the *ASP*-tree, so that a node can count at most $\alpha \times n$ points if it is not a unit (minimal solution) cell. *ASP*-tree can adapt to the changes in the distribution of the stream by dynamically splitting and merging cells so that high density regions have fine cell subdivisions while a rougher subdivision is sufficient for low density regions. Given α, it is shown in [14] that the tree size is at most $\mathcal{O}(\frac{1}{\alpha})$ with amortized update time $\mathcal{O}(\log(\frac{1}{\alpha}))$.

Same as the standard quadtree, given a search region R, the number of points within R can be estimated by $\sum_{e \in \mathcal{T}} n.v \times \frac{Area(e.c \cap R)}{Area(e.c)}$, where $Area(R)$ denotes the area of a region R. Note that a more sophisticated *ASP*-tree structure with a $\log(J^2)$ factor increase in space is proposed in [14] to achieve ϵ relative error guarantee for axis-aligned range query with space $\mathcal{O}(\frac{n}{\epsilon s}\log(J^3))$, where s represents the number of objects within R. Nevertheless, we adopt the standard version in this paper, since it is more space efficient and can consistently achieve good performance for 2-dimensional data in practice.

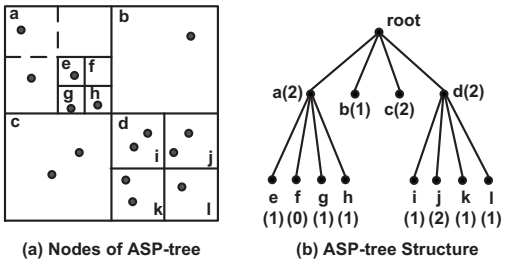


Figure 2: *ASP*-tree

2.3.2 KMV Synopses

The k minimal values (**KMV**) technique is first proposed by Bar-Yossef *et al.* in [2] to estimate the number of distinct values in a data stream. Suppose h is a pair-wise independent hash function which randomly maps the values onto the range $[0, 1]$ and $h(v_i) \neq h(v_j)$ for any two different values v_i and v_j. A KMV synopses of a set $\mathcal{D}$ of values, denoted by $\mathcal{L}_\mathcal{D}$, keeps k smallest hash values of the elements in $\mathcal{D}$. Then the number of distinct values in $\mathcal{D}$, denoted by $\hat{D}$, can be estimated by $\hat{D} = \frac{k}{\mathcal{L}^{(k)}}$ where $\mathcal{L}^{(k)}$ is k-th smallest hash value. In [3], Bayer *et al.* systematically investigate the problem of distinct value estimation under multi-set operations. They show that $\frac{k-1}{\mathcal{L}^{(k)}}$ is an unbiased estimator of $\hat{D}$, which is used for multi-set operations as follows.

Union Operation: Consider two sets $\mathcal{A}$ and $\mathcal{B}$, with corresponding KMV synopses $\mathcal{L}_\mathcal{A}$ and $\mathcal{L}_\mathcal{B}$ of size k_A and k_B, respectively. In this paper, we use $\cup_m$ ($\cap_m$) to denote the union (intersect) operation which removes the duplicate values. In [3], $\mathcal{L}_\mathcal{A} \oplus \mathcal{L}_\mathcal{B}$ is used to denote the set comprising the k smallest distinct hash values in $\mathcal{L}_\mathcal{A} \cup_m \mathcal{L}_\mathcal{B}$ where $k = min(k_A, k_B)$. Then $\mathcal{L} = \mathcal{L}_\mathcal{A} \oplus \mathcal{L}_\mathcal{B}$ is the KMV synopses of $\mathcal{A} \cup \mathcal{B}$. The number of distinct values in $\mathcal{A} \cup \mathcal{B}$, denoted by $D_\cup$, can be estimated as follows.

$$\hat{D}_\cup = \frac{k-1}{\mathcal{L}^{(k)}} \tag{1}$$

Intersection Operation: Same as the union operation, we use the KMV synopses $\mathcal{L} = \mathcal{L}_\mathcal{A} \oplus \mathcal{L}_\mathcal{B}$ where $k = min(L_A, L_B)$. Let $K_\cap$ denote the number of common distinct hash values in $\mathcal{L}$ regarding $\mathcal{L}_\mathcal{A}$ and $\mathcal{L}_\mathcal{B}$, i.e., $K_\cap = |\{v \in \mathcal{L} : v \in \mathcal{L}_\mathcal{A} \cap_m \mathcal{L}_\mathcal{B}\}|$. We can estimate the number of distinct values in $\mathcal{A} \cap \mathcal{B}$, denoted by $D_\cap$, as follows.

$$\hat{D}_\cap = \frac{K_\cap}{k} \times \frac{k-1}{\mathcal{L}^{(k)}} \tag{2}$$

As shown in [3], Equation 1 and 2 can be easily applied to multi-set operations where $\mathcal{L} = \mathcal{L}_{A_1} \oplus \ldots \mathcal{L}_{A_m}$ and $k = min(k_{A_1}, \ldots, k_{A_m})$.

2.3.3 Bayesian Networks (BNs)

In recent years, probabilistic graphical models have been widely used in the literature to capture the *conditional independence* among attributes in the distribution, and hence allow us to specify the joint distributions over high dimensional space compactly. Like many selectivity estimation algorithms (e.g., [23]), we adopt the Bayesian network (BN), denoted by $\mathcal{B}(\mathcal{G}, \mathcal{P})$, which consists of two components: a network structure $\mathcal{G}$ and model parameters $\mathcal{P}$. Specifically, the network structure $\mathcal{G}$ is a directed acyclic graph $\mathcal{G}(\mathcal{V}, \mathcal{E})$ where each vertex $X \in \mathcal{V}$ represents a random variable X and an edge $X_i \rightarrow X_j \in \mathcal{E}$ denotes that the value of X_j is (stochastically) influenced by the value of X_i. Let $Pa(X)$ denote the parent vertices of X in $\mathcal{G}$, X is dependent on the random variables in $Pa(X)$, but conditional independent of the random variables in non-descendant vertices, i.e., vertices which are not reachable by X following the edges in $\mathcal{G}$. The model parameters ($\mathcal{P}$) describe the statistical relationship between each vertex and its parents. A *conditional probability distribution* (CPD) $P_\mathcal{B}(X_i|Pa(X_i))$ is calculated for each random variable X_i in $\mathcal{G}$ which describes the distribution of X_i for any given possible assignment of it parents. We say X is an **independent vertex** if $Pa(X) = \emptyset$, otherwise X is a **dependent vertex**. Informally, the correlations among the attributes are captured by the conditional probability distributions of the dependent vertices, i.e., $\{P_\mathcal{B}(X_i|Pa(X_i)\}$ where $Pa(X_i) \neq \emptyset$. For presentation simplicity, $P_\mathcal{B}(X)$ is abbreviated to $P(X)$ whenever there is no ambiguity.

Given a Bayesian network $\mathcal{B}$, we can derive the joint probability distribution of a set of random variables based on the chain rule; that is,

$$P_\mathcal{B}(X_1, \ldots, X_m) = \prod_{i=1}^{m} P_\mathcal{B}(X_i|Pa(X_i)) \tag{3}$$

Consequently, the selectivity of a given query can be estimated based on the BN inference techniques [23].

3. RC AND DVs BASED ESTIMATORS

In this section, we present two algorithms based on RC estimator techniques and DVs estimator techniques, respectively. Specifically, Section 3.1 adopts the *ASP*-tree [14] technique to estimate the cardinality of the spatial keyword query using independence assumption. Section 3.2 investigates how to apply the KMV synopses technique. Section 3.3 analyzes the performance of two algorithms.

3.1 RC Estimator based Algorithm (RC-E)

The problem of range counting estimation on streaming spatial data has been intensively studied. In this paper, we adopt the *ASP*-tree structure because it has good performance for range counting estimation on 2-dimensional streaming spatial data.

Maintaining *ASP*-trees. At initial stage, we continuously maintain an *ASP*-tree for each term t_i, denoted by $\mathcal{T}_i$, and the corresponding split threshold α_i is set to 0. When

the occupied space reaches the space budget, the space allocated for each term t_i is proportional to its frequency n_i by increasing the split threshold of $\mathcal{T}_i$. In many applications, the number of distinct terms might be extremely large. For instance, there are around 1 million distinct terms in Twitter dataset in the experiments. Nevertheless, it has been widely observed that the frequencies of the terms follow the *power law* distribution, which implies that a large portion of terms have very low frequencies. Since no existing method can achieve a good accuracy in terms of relative error for spatial range queries with very small cardinality, we only maintain *ASP*-tree for terms with frequencies at least ϵn^* in our implementation, where n^* is the total frequencies of the terms seen so far.

Selectivity Estimation. RC estimator based algorithm, namely RC-E, is simple and intuitive. Suppose an *ASP*-tree $\mathcal{T}_i$ is constructed for all objects containing the term t_i, and $\hat{A}_i$ denotes the size estimation of range query $q.R$ on $\mathcal{T}_i$. With independence assumption, we have the following formula for the selectivity estimation of query q, denoted by $\hat{A}$, where n is the total number of objects.

$$\hat{A} = n \times \prod_{t_i \in q.\mathcal{V}} \frac{\hat{A}_i}{n} \tag{4}$$

For each term t_i without *ASP*-tree, a counter n_i is used to record its frequency, then $\hat{A}_i$ is estimated by $n_i \times \rho$, where ρ is the ratio between the area of $q.R$ and the space.

Time Complexity Analysis. For the given search region, the time cost to estimate $\hat{A}_i$ in Equation 4 corresponds to a range search on the quadtree, which is very efficient in practice. As shown in Section 2.3.1, when an object o arrives, the amortized update time cost is $\mathcal{O}(\log(\frac{1}{\alpha_i}))$ for each term $t_i \in o.\mathcal{V}$ where α_i is the split threshold of the *ASP*-tree $\mathcal{T}_i$.

3.2 DVs Estimator based Algorithm (DVs-E)

Given a query q, $\mathcal{D}_\cap(q)$ denotes the set of objects satisfying both spatial and keywords constraints, and $\mathcal{D}_i(q)$ represents the objects with term t_i, which resides in $q.R$. Clearly, we have $\mathcal{D}_\cap(q) = \bigcap_{t_i \in q.\mathcal{V}} \mathcal{D}_i(q)$. Therefore, it is intuitive to take advantage of the DVs estimators which can effectively support the multi-set operators such as union and intersection. In this paper, we adopt the representative DVs estimator, KMV synopses [3]. Nevertheless, the algorithm can be easily modified for other min-wise hashing based DVs estimators [18].

Algorithm 1: **update KMV synopses** (o)

```
   Input  : o : a new arriving object
   Output: Updated KMV synopses L
1  v := h(o) ;
2  if v < v_τ then
3      L := L ∪ o;
4      for each t_i ∈ o.V do
5          L_i := L_i ∪ o ;
6      τ := τ + 1 ;
7  while the size of L exceeds the memory budget B do
8      o_j ← the object with the τ-th smallest hash value ;
9      L := L \ o_j;
10     for each t_i ∈ o_j.V do
11         L_i := L_i \ o_j;
12     τ := τ − 1;
```

Maintaining KMV synopses. For each term t_i, $\mathcal{L}_i$ denotes the KMV synopses (i.e., sample objects[2]) of $\mathcal{D}_i$ with k_i smallest hash values. Throughout, all synopses are generated with the same hash function h defined in Section 2.3.2. We continuously maintain τ objects with smallest hash values, denoted by $\mathcal{D}_\tau$, so that the total space consumed is bounded by the memory budget. Then we have $\mathcal{L}_i = \{o : t_i \in o.\mathcal{V} \text{ and } o \in \mathcal{D}_\tau\}$. Algorithm 1 illustrates the details of the KMV synopses maintenance. The threshold hash value v_τ is set to ∞ before $\mathcal{L}$ exceeds the space budget B at the first time and then v_τ is set to the τ-th smallest hash values in $\mathcal{L}$, i.e., the largest hash values in $\mathcal{L}$. Suppose there are n objects in the stream seen as far, each object is chosen with probability $\frac{\tau}{n}$ and hence the expected size of $\mathcal{L}_i$ is $\sum_{o \in \mathcal{D}} Pr(t_i \in o.\mathcal{V}) \times \frac{\tau}{n} = freq(t) \times \frac{\tau}{n}$ which indicates that the resource allocation among KMV synopses of the terms is proportional to their frequencies. A sampled object o will be organized by $\{\mathcal{L}_i\}$ for each term $t_i \in o.\mathcal{V}$. Particularly, we adopt the grid based inverted indexing technique [6] to organize sample objects where a posting list of objects in $\mathcal{L}_i$ is maintained for each term t_i, and sample objects in $\mathcal{L}_i$ are sorted by their Z-orders to facilitate the range search. Consequently, for a given query q, we can quickly retrieve $\mathcal{L}_i(q)$ for each query keyword t_i.

Algorithm 2: **DVs-E** (q, $\mathcal{L}$)

```
   Input  : q : query, L : KMV synopses of D
   Output : the selectivity estimation of q
1  L(q) := ∅ ;
2  for each query keyword t_i ∈ q.V do
3      L_i(q) := objects of L_i within q.R;
4      L(q) := L(q) ∪_m L_i(q);
5  k := |L(q)|;
6  K_∩ ← the number of representative samples in L(q);
7  return Â := (K_∩ / k) × (k − 1) / L(q)^(k)
```

Selectivity Estimation. Algorithm 2 illustrates the details of the KMV synopses based selectivity estimation. Given a query q and the continuously maintained KMV synopses $\mathcal{L}$ on the stream $\mathcal{D}$, for each query keyword t_i, $\mathcal{L}_i(q)$ retrieves the objects in $\mathcal{L}$ each of which contains term t_i and resides in $q.R$. Then the union of the $\mathcal{L}_i(q)$ for $t_i \in q.\mathcal{V}$, denoted by $\mathcal{L}(q)$, can be used for the selectivity estimation of q according to Equation 1. We set $k = |\mathcal{L}(q)|$ at Line 5, in contrast to setting $k = min(\{\mathcal{L}_i(q)\})$ for $t_i \in q.\mathcal{V}$ in Section 2.3.2. This enhances the accuracy of the estimation because a larger k value (i.e., synopses size) usually leads to higher accuracy. In this paper, we say an object o is a **representative sample** of $\mathcal{L}$ if it is chosen by $\mathcal{L}$ and satisfies the query constraints, i.e., $o \in \mathcal{D}(q)$ and $o \in \mathcal{L}$. We use $K_\cap$ to denote the number of representative samples in $\mathcal{L}(q)$. Clearly, $K_\cap$ is proportional to the query result size, i.e., $E(K_\cap) = \frac{A}{n} \times |\mathcal{L}|$ since the hash value are randomly chosen for each object. Then Line 7 estimates the selectivity of q according to Equation 2.

Algorithm Correctness. Lemma 1 below shows that KMV synopses obtained in Algorithms 2 is valid, and hence the correctness of Algorithm 2 immediately follows.

Lemma 1. *The KMV synopses $\mathcal{L}(q)$ obtained in Algorithm 2 is valid.*

PROOF. Suppose the size of $\mathcal{L}(q)$ is k, and v_k represents the k-th smallest hash value in $\mathcal{L}(q)$. We say $\mathcal{L}_q$ is valid

[2]To support spatial search on KMV synopses, we keep sample objects instead of their hash values.

if v_k is the k-th smallest hash value regarding all objects containing any query keyword. That is, if there is an object o with $h(o) < v_k$ and o contain at least one query keyword, o must be included by $\mathcal{L}(q)$. Let v_τ be the τ-th smallest hash value for all objects in $\mathcal{D}$. An object o associated with term t_i will be stored in $\mathcal{L}_i$ if $h(o) \leq v_\tau$. It is immediate that o will be kept in $\mathcal{L}_i$ if $h(o) \leq v_k$ since $v_k \leq v_\tau$. Consequently, $\mathcal{L}(q)$ obtained in Algorithm 2 is correct. □

Time Complexity Analysis. The query response time of Algorithm 2 is $\mathcal{O}(m \times L)$ in the worse case where L is the average size of the posting lists. Nevertheless, the retrieval of $\mathcal{L}(q)$ is efficient in practice since sample objects for each term are organized by the grid index. Regarding the maintainable cost, it takes $\mathcal{O}(|o.\mathcal{V}| \times \log(L))$ time when an object is chosen or rejected from $\mathcal{L}$.

3.3 Performance Comparison

In this subsection, we compare the performance of RC-E and DVs-E algorithms, and show their advantages and inherent shortcomings w.r.t the problem studied in this paper.

3.3.1 Size Estimation on Single Query Keyword

As shown in Section 2.3.1, if there is only one query keyword t_i and objects are within the space $[0, J]^2$, RC-E algorithm can achieve ϵ relative error for an axis-aligned range query with space $\mathcal{O}(\frac{n_i}{\epsilon A} \log J^3)$. Recall that n_i and A represent the number of objects with query keyword t_i and the cardinality of the query q, respectively. On the other hand, according to [2], $\mathcal{O}((\frac{n_i}{\epsilon A})^2 \log \frac{1}{\delta})$ space is required for DVs-E to achieve ϵ relative error guarantee where δ is the confidence value. This implies that *ASP*-tree is more effective for the size estimation on single query keyword. Indeed, our empirical study demonstrates that the estimate accuracy of RC-E significantly outperforms that of DVs-E for single query keyword. This is not surprising because *ASP*-tree is the summary technique dedicated for the range counting over streaming spatial data.

3.3.2 Size Estimation on Multiple Query Keywords

The performance of RC-E relies on the independence assumption among query keywords. Unfortunately, real-life data consistently violates this assumption and hence the performance of RC-E tends to be highly inaccurate in our empirical study when the number of query keywords grows. Furthermore, an interesting observation in this paper is that the correlations among query keywords can hardly alleviate this problem if the correlations are obtained from all objects in the space. This motivates us to exploit the "locality" of probabilistic influence among query keywords in Section 4.

On the other hand, the estimation quality of DVs-E consistently outperforms that of RC-E because DVs estimators can accurately estimate the set intersection. However, it has been widely observed that the variance of the DVs estimator is very large when there is insufficient number of representative samples. For instance, Neustar sets up a webpage [3] to evaluate the performance of KMV synopses, which demonstrates that the performance of KMV is quite unstable until the number of representative samples reaches a certain threshold. Similar observation is also reported in [18] for min-wise hashing based DVs estimators. As shown in our empirical study, DVs-E suffers from this problem because the number of representative sample (i.e., sample objects surviving both spatial and keyword constraints) could be small even a large number of objects are sampled and the search region is moderately large.

[3] `http://research.neustar.biz/2012/07/09/sketch-of-the-day-k-minimum-values/`

4. ESTIMATE USING LOCAL CORRELATIONS

In this section, we present a novel computing paradigm based on the local correlations of the query keywords. Firstly, Section 4.1 introduces the motivation of our technique based on two key observations of local correlations among query keywords, followed by the details of the A^2SP-tree structure which effectively integrates *ASP*-tree and KMV synopses techniques in Section 4.2. Then Section 4.3 presents the selectivity estimation algorithm based on BNs which can capture the local correlations among query keywords. Section 4.4 introduces how to learn a local BN on-the-fly for the given spatial keyword query. Finally, Section 4.5 conducts performance analysis.

4.1 Motivation

A large body of works (e.g., [23]) have shown that the accuracy of the selectivity estimation can be significantly improved by exploiting correlations among attributes for a variety of queries because the correlation is inherent in many real-life datasets. This motivates us to propose new computing paradigm by exploiting local correlations among query keywords. Below is our first key observation.

Observation 1. *Many terms are highly correlated in spatio-textual objects. Moreover, correlations of these terms exhibit the "locality" property.*

The "locality" of the correlations is a common phenomenon for spatio-textual objects in many applications. For instance, two terms "chips" and "fish" may co-occur frequently in tweets introducing local food in Sydney, while they may become irrelevant in Dubai. Batkid event mentioned in Section 1 is another example. Thus, we aim to propose a new computing paradigm to derive selectivity estimation by exploiting the local correlations among query keywords w.r.t the search region. In this paper, we adopt the popular probabilistic graphical model, Bayesian networks (BNs), to capture the local correlations.

The stream $\mathcal{D}$ can be modeled as a relational table where each term $t \in \mathcal{V}$ is an attribute and an object is a record with $|\mathcal{V}|$ attributes. An attribute associated with term t_i corresponds to a binary random value, denoted by T_i. Given a region R with N objects and a set of terms $\mathcal{X} = \{T_1, \ldots, T_m\}$, we say a BN is a **local Bayesian Network** of $\mathcal{D}$ w.r.t R and $\mathcal{X}$ if it is learned from the objects within R and each term in $\mathcal{X}$ corresponds to a vertex in the network structure.

For a term t_i, $P(T_i = 1)$ denotes the probability that an object within R contains the term t_i where $P(T_i = 1) = \frac{N_i}{N}$, and N_i is the frequency of t_i among N objects. Similarly, given a set of terms $\mathcal{X} = \{T_1, \ldots, T_m\}$, $P(\mathcal{X} = 1)$ represents the probability that an object within R contains all terms in $\mathcal{X}$. Given the local BN of the query, we can immediately derive the selectivity estimation based on the chain rule (Equation 3). In particular, let $\mathcal{X}$ represent a set of variables for query keywords and θ denote the probability that an object within the search region is among the query results, we have $\theta = P(\mathcal{X} = 1) = P(T_1 = 1, \ldots, T_m = 1) = \prod_{X_i \in \mathcal{X}} P(X_i = 1 | Pa(X_i) = 1)$. Then we have $\hat{A} := N \times \theta$, where N is the number of objects within the search region.

Example 2. *Given a set of spatio-textual objects in Figure 3(a), and a query q with $q.\mathcal{V} = \{t_1, t_2, t_3\}$ and $q.R = R_1$. Figure 3(b) illustrates the local BN $\mathcal{B}_1$ regarding q where T_2*

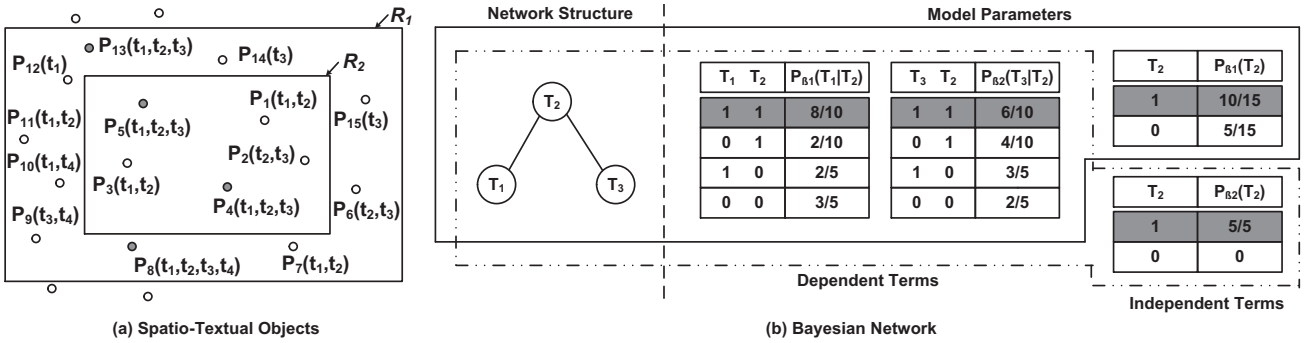


Figure 3: Example for Local Bayesian Networks

is an independent term (vertex), while T_1 and T_3 are dependent terms (vertices). $N = 15$ and then we have $\hat{A} = N \times P_{\mathcal{B}_1}(T_1 = 1|T_2 = 1) \times P_{\mathcal{B}_1}(T_3 = 1|T_2 = 1) \times P_{\mathcal{B}_1}(T_2 = 1) = 15 \times 8/10 \times 6/10 \times 10/15 = 4.8$.

REMARK 1. *To support the selectivity estimation of spatial keyword search we only need materialize $P(T = 1|Pa(T) = 1)$ for the CPDs of query keywords $\{T\}$. E.g., shaded rows of CPDs in Figure 3(b).*

It is a great challenge to capture local correlations among the terms given the large number of distinct terms and arbitrary search regions, not to mention the massive streaming objects arriving at a rapid rate. Clearly, it is infeasible to exactly compute the local BN in the streaming context. In this paper, we introduce an augmented adaptive space partition tree (A^2SP-tree for short) which effectively integrates the strengths of ASP-tree and KMV synopses techniques. Section 3.3 shows that ASP-tree and KMV synopses are good at range counting and multi-set operations, respectively. While both two functions are critical to the learning of local BN as shown in Section 4.4.

As shown in our empirical study, the performance of KMV synopses is unreliable when there is insufficient number of representative samples, which may lead to poor learning results of BN. This motivates us to develop *local boosting* technique based on the following key observation.

Observation 2. *The local correlations among the terms are likely to be preserved if the search region is enlarged within the same local area.*

Observation 2 is quite intuitive because correlations among the terms, which are captured by CPDs of dependent terms (vertices), are unlikely to change dramatically within a local area. The effectiveness of our *local boosting* approach in empirical study verifies this observation in real-life datasets. Formally, suppose $\mathcal{B}_1$ and $\mathcal{B}_2$ are two local BNs obtained from regions R_1 and R_2 ($R_2 \subset R_1$), respectively. We say local correlations w.r.t R_2 is well preserved in R_1 if the two BNs share the same graph structure $\mathcal{G}$ and for any dependent vertex T_i in $\mathcal{G}$ we have $P_{\mathcal{B}_1}(T_i|Pa(T_i)) \approx P_{\mathcal{B}_2}(T_i|Pa(T_i))$. Recall that $P_{\mathcal{B}_1}(T_i|Pa(T_i))$ and $P_{\mathcal{B}_2}(T_i|Pa(T_i)$ denote the CPDs of the term T_i in $\mathcal{B}_1$ and $\mathcal{B}_2$, respectively. Note that this property is unlikely to hold for independent terms (vertices) as the probabilistic distribution of a term t_i (i.e., $P(T_i)$) may change dramatically within a local area.

Example 3. *Suppose $\mathcal{B}_1$ and $\mathcal{B}$ are two local BNs learned based on R_1 and R_2 in Figure 3(a), respectively, and they share the same network structure as shown in Figure 3(b). Regarding two dependent vertex T_1 and T_3, we have $P_{\mathcal{B}_1}(T_1 = 1|T_2 = 1) = 8/10$, $P_{\mathcal{B}}(T_1 = 1|T_2 = 1) = 4/5$, $P_{\mathcal{B}_1}(T_3 = 1|T_2 = 1) = 6/10$, and $P_{\mathcal{B}}(T_3 = 1|T_2 = 1) = 3/5$. Regarding independent vertex T_2, we have $P_{\mathcal{B}_1}(T_2 = 1) = 10/15$ and $P_{\mathcal{B}}(T_2 = 1) = 1$.*

The key idea of *local boosting* is to carefully expand the search region such that there are sufficient representative samples to accurately learn correlations (i.e., conditional dependencies) among query keywords. Specifically, we apply *local boosting* to learn a boosted BN $\mathcal{B}(\mathcal{G}, \mathcal{P})$ w.r.t the query q with following four steps.

- **Step 1**. Find a region R with $q.R \subset R$ in which there are sufficient representative samples.
- **Step 2**. Learn graph structure $\mathcal{G}$ based on R and query keywords.
- **Step 3**. Learn model parameter $\mathcal{P}$ for dependent terms (query keywords) $\{T_i\}$: derive $P_{\mathcal{B}}(T_i = 1|Pa(T_i) = 1)$ against R.
- **Step 4**. Back to the region $q.R$ and learn model parameter $\mathcal{P}$ for independent terms (query keywords) $\{T_j\}$: $P_{\mathcal{B}}(T_j = 1) = n_j(q)/N$ where N is the number of objects within $q.R$, and $n_j(q)$ is the frequency of t_j among these N objects. Note that we can accurately estimate $n_j(q)$ based on $\mathcal{T}_j$ since only one query keyword is involved.

Following is an example about how to construct a boosted BN and derive selectivity estimation.

Example 4. *In Figure 3, suppose $q.R = R_2$ and $q.\mathcal{V} = \{t_1, t_2, t_3\}$. We may first learn the local BN $\mathcal{B}_1$ based on the enlarged region R_1. Then, for the independent vertex T_2, we replace $P_{\mathcal{B}_2}(T_2 = 1)$ with $P_{\mathcal{B}_1}(T_2 = 1)$ which is calculated against R_2 as shown in Figure 3(b). Thus, we have $\hat{A} = N \times P_{\mathcal{B}_1}(T_1 = 1|T_2 = 1) \times P_{\mathcal{B}_1}(T_3 = 1|T_2 = 1) \times P_{\mathcal{B}_2}(T_2 = 1) = 5 \times 8/10 \times 6/10 \times 5/5 = 2.4$.*

4.2 A²SP Structure

The A^2SP-tree of the stream $\mathcal{D}$, denoted by $\mathcal{S}$, consists of two components: a KMV synopses $\mathcal{L}$ of $\mathcal{D}$ and a set $\mathcal{T}$ of ASP-trees. Given a memory budget B and a value λ with $0 < \lambda < 1$, we allocate $\lambda \times B$ and $(1 - \lambda) \times B$ spaces to $\mathcal{L}$ and $\mathcal{T}$, respectively. Section 6 will investigate the setting of λ. A KMV synopses introduced in Section 2.3.2 is continuously maintained for streaming spatio-textual objects. Meanwhile, we also keep an ASP-tree for a term t_i, denoted by $\mathcal{T}_i$, if $\mathcal{L}_i \neq \emptyset$. Instead of keeping objects in $\mathcal{L}$ and $\mathcal{T}$ separately, we maintain a set of pointers for each KMV sample object o and its corresponding cells in $\mathcal{T}$, i.e., the cells which count o. Consequently, a cell and its corresponding sample

objects can be accessed simultaneously to speed up the computation. Figure 4 depicts an example of A^2SP-tree where objects sampled by KMV synopses $\mathcal{L}$ are organized by corresponding ASP-trees.

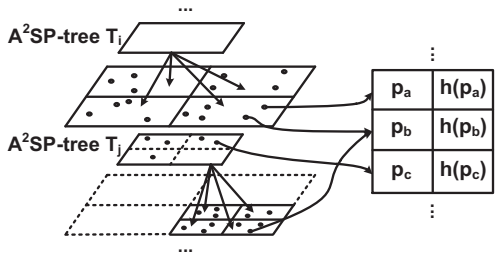


Figure 4: A^2SP-tree Structure

4.3 Selectivity Estimation

Algorithm 3 illustrates the details of the selectivity estimation based on A^2SP-tree. If there is only one query keyword t or one of the query keyword does not have the ASP-tree due to its low frequency, Line 2 simply applies the RC-E Algorithm introduced in Section 3.1. Otherwise, we follow the local correlation based computing paradigm. $\mathcal{L}(q)$ retrieves the union of the KMV synopses for all query keywords (Line 3). We use $K_\cap$ to denote the number of representative samples in $\mathcal{L}(q)$. In this paper, the *local boosting* strategy is applied at Line 6 if $K_\cap < K$, which implies the performance of $\mathcal{L}(q)$ might be unreliable. Otherwise, *local boosting* is not necessary (Line 8). We investigate the setting of boosting threshold K in Section 6. Moreover, We will introduce the learning of local BN w.r.t q in Section 4.4. Lines 10-14 compute the selectivity estimation of q by utilizing the joint distribution of the query keywords as discussed in Section 4.1. Note that, we only consider the objects containing at least one query keyword for the construction of local BN, and the total number of objects involved, denoted by N, can be estimated by Equation 1.

Algorithm 3: **BN-E** (q, $\mathcal{S}$)

Input : q : query object, $\mathcal{S}$: A^2SP-tree of $\mathcal{D}$
Output : the selectivity estimation of q
1 **if** $|q.\mathcal{V}| = 1$ or $\mathcal{T}_i = \emptyset$ for certain $t_i \in q.\mathcal{V}$ **then**
2 **return** $\hat{A}$ based on RC-E
3 $\mathcal{L}(q) \leftarrow$ objects in $\mathcal{L}$ falling in $q.R$;
4 $K_\cap \leftarrow$ the number of representative samples in $\mathcal{L}(q)$;
5 **if** $K_\cap < K$ **then**
6 Learn local BN $\mathcal{B}$ for q **with** *local boosting*;
7 **else**
8 Learn local BN $\mathcal{B}$ for q **without** *local boosting* ;
9 $\theta := 1$;
10 **for** each term t_i in $q.\mathcal{V}$ **do**
11 **if** $Pa(T_i) = \emptyset$ **then**
12 $\theta := \theta \times P(T_i = 1)$;
13 **else**
14 $\theta := \theta \times P(T_i = 1|Pa(T_i) = 1)$;
15 $N \leftarrow$ number of objects within $q.R$ containing at least one query keyword;
16 **return** $\hat{A} := N \times \theta$

4.4 Learn Bayesian Networks On-the-Fly

In this subsection, we first investigate how to enlarge the search region when the *local boosting* is required. Then we propose efficient local BN learning algorithm for a given region R, which might be the query region (i.e., $R = q.R$) or an enlarged search region.

For a given region R, we use $\mathcal{L}(R)$ to represent the sampled KMV objects in $\mathcal{L}$ within R, and K_R to denote the number of representative samples in $\mathcal{L}(R)$. Note that we have $R = q.R$ and $\mathcal{L}(R) = \mathcal{L}(q)$ if the local boosting is not necessary (i.e., $K_R \geq K$ in Algorithm 3). Otherwise, we will enlarge $q.R$ to a new region R as follows.

Enlarge the Search Region (R). Ideally, we want to find a minimal region R such that K_R is larger than K, where K is the threshold for *local boosting*. However, it is cost-prohibitive to find such an optimal R and hence we resort to a heuristic method, which tends to be quite effective in practice. Since all ASP-trees can be regarded as a standard quadtree with height $\log(J)$, denoted by $\mathcal{Q}$. We first find the minimal quadtree cell R' of $\mathcal{Q}$ which contains $q.R$. Starting from R', we traverse along the path from R' to the root of $\mathcal{Q}$ until we reach the root or stop at a cell R with sufficient number of samples in $\mathcal{L}(R)$ (i.e., $K_R > K$).

Then we learn the network structure and model parameters of $\mathcal{B}$ regarding the region R and the query keywords as follows.

Learn Network Structure ($\mathcal{G}$). We adopt the popular *mutual information* metric [8] to evaluate the dependence between two random variables X_i and X_j, dentoed by $I(X_i, X_j)$, where

$$I(X_i, X_j) = \sum_{X_i} \sum_{X_j} P(X_i, X_j) \log(\frac{P(X_i, X_j)}{P(X_i)P(X_j)}) \qquad (5)$$

Let N_i and N_j denote the number of terms $t_i, t_j \in q.\mathcal{V}$ in the region R, respectively, which can be obtained by issuing range counting queries on ASP-trees $\mathcal{T}_i$ and $\mathcal{T}_j$. $N_\cup$ ($N_\cap$) represents the number of distinct objects in the union (intersection) of two sets, which is derived based on the KMV synopses of R. Then we obtain the joint probability distribution $P(X_i, X_j)$ using $N_\cap$, $N_\cup$, N_i, N_j and N.

By computing the pair-wise mutual information for each pair of query keywords, we come up with a complete graph with $|q.\mathcal{V}|$ vertices and the weight of an edge between X_i and X_j is $I(X_i, X_j)$. Same as [8], the structure $\mathcal{G}$ of $\mathcal{B}$ corresponds to the maximal spanning tree of this complete graph.

Learn Model Parameters ($\mathcal{P}$). With similar rationale, we compute the model parameters (i.e., CPDs) based on the KMV synopses and ASP-trees of the region R. Note that we only need to compute the probability $P(T = 1|Pa(T) = 1)$ for each independent vertex $T \in \mathcal{G}$. As shown in Section 4.1, if R is an enlarged region we need to compute the probability $P(T = 1)$ for all independent vertices $T \in \mathcal{G}$ against $q.R$ by issuing range counting queries on their corresponding ASP-trees.

4.5 Performance Analysis

The dominant costs of Algorithm 3 comes from the retrieval of the KMV synopses for query q and the learning of the local BN. Although $\mathcal{O}(|\mathcal{L}|)$ time is required to retrieve KMV synopses in the worst case, it is efficient in practice because sampled KMV objects are organized by ASP-trees. Let l denote the number of sample objects in KMV synopses $\mathcal{L}(R)$, it takes at most $\mathcal{O}(l)$ time to calculate $N_\cup$ and $N_\cap$ for each pair of vertices. The cost to learn the structure $\mathcal{G}$ is $\mathcal{O}(m^2 \times l + m \times b)$ in the worst case where b is the cost to estimate N_i for a term t_i against the ASP-tree $\mathcal{T}_i$. Recall m is the number of query keywords. Then it

takes $\mathcal{O}(m^2)$ time to compute the pairwise mutual information and to generate the maximal spanning tree. As for the learning of the parameters, it takes at most $\mathcal{O}(m)$ time to materialize the probability $P(T = 1|Pa(T) = 1)$ for each query keyword. Consequently, time cost for Algorithm 3 is $\mathcal{O}(m^2 \times l + m \times b + |\mathcal{L}|)$ in the worst case, which is quite efficient in practice because the number of query keywords (m) in spatial keyword search is usually rather small. Moreover, by utilizing the tree structure of A^2SP-tree, we may efficiently retrieve KMV synopses, and support the range counting against the query region $q.R$ as well as the enlarged region R.

5. EXTENSIONS

In this section, we discuss the extension of our techniques to support sliding window model and boolean spatial keyword queries.

5.1 Extension for Sliding Window Model

In many real applications, users only care the most recent data, e.g., tweets within last 24 hours or the most recent 1 million tweets. Thus, it is desirable to extend our estimation techniques to support sliding window model. In this paper, we focus on the fixed size sliding window model, where the most recent n objects, denoted by $\mathcal{D}_n$, are considered for queries.

The key is to continuously maintain the KMV synopses and *ASP*-trees over sliding window. The maintenance of *ASP*-tree over sliding window has been investigated in [14]. Below, we briefly introduce how to maintain KMV synopses $\mathcal{L}$. Since sample objects in $\mathcal{L}$ may expire, we cannot simply keep τ objects with the smallest hash values in the sliding window model. Consequently, we have to keep a candidate set $\mathcal{L}_c$ for objects which may contribute to $\mathcal{L}$ in the future. At first glance, we have $\mathcal{L}_c = \mathcal{D}/\mathcal{L}$. Fortunately, the size of $\mathcal{L}_c$ can be significantly reduced by utilizing the concept of *skyband* [19]. We say an object o_i dominants another object o_j, if o_i arrives earlier than o_j and $h(o_i) < h(o_j)$. It is safe to prune an object if it is dominated by at least τ objects in $\mathcal{D}_n$. Thus, we only need to continuously maintain the skyband of current sliding window, i.e., objects survived from the above dominance check.

The maintenance of A^2SP-tree is immediate since it consists of *ASP*-trees and KMV synopses.

5.2 Boolean Spatial Keyword Queries

Our *local correlation* based algorithm can be easily adopted to support boolean spatial keyword queries where conjunctions (ANDs), disjunctions (ORs) and negations (NOTs) are considered, because we can materialize the full joint probability distribution of the query keywords according to the local BN. More specifically, given a boolean keyword query q, and a set $\mathcal{X}$ of binary random variables for query keyword t_1, ..., t_m where $\mathcal{X} = \{T_1, \ldots, T_m\}$. An instance of $\mathcal{X}$, denoted by X, is an assignment of $\{T_i = c_i\}$ where c_i equals 0 or 1. We say an instance X satisfies the query if $\{T_i = c_i\}$ meets the query keyword constraint. For instance, suppose the query keyword constraint is $(t_1 \vee t_2) \wedge (t_1 \wedge t_3)$ in the example of Figure 3, then two instances of $\mathcal{X}$, X_a and X_b, satisfy the query where $X_a = \{T_1 = 1, T_2 = 0, T_3 = 1\}$ and $X_a = \{T_1 = 1, T_2 = 1, T_3 = 1\}$. The selectivity of q can be estimated by $N \times (P(\mathcal{X} = X_a) + P(\mathcal{X} = X_b))$, where N is the number of objects in $q.R$. Clearly, we only need to materialize the probabilities for the instances which satisfy the query.

To support boolean queries, we need to transfer the query to disjunctive normal form (DNF) for RC-E and DVs-E. Each component c_{dnf} in DNF, in the form of $(t_{i1} \wedge ... \wedge t_{ij} \wedge ... \wedge \neg t_{ij+1} \wedge ... \wedge \neg t_{ik})$, is calculated as $|(t_{i1} \wedge ... \wedge t_{ij}| - |(t_{i1} \wedge ... \wedge t_{ik})|$, where $t_{ij} \in q.\mathcal{V}$. Then we can follow the sets *inclusive-exclusive* formula to answer query q by a series of addition/deduction of sets intersection.

Property	GN	TW	Cars
# objects	$2.2M$	$11.5M$	$2.25M$
vocabulary size	$208K$	$1.02M$	$81K$
avg. # keywords	6.75	9.3	26

Table 2: Dataset Details

6. PERFORMANCE EVALUATION

In this section, we present the results of a comprehensive performance study to evaluate the efficiency and effectiveness of the proposed techniques in this paper.

6.1 Experiment Setup

As there is no previous work for the problem of selectivity estimation on streaming spatio-textual objects, we evaluate the following algorithms proposed in this paper.

- **RC-E**. *ASP*-tree based algorithm proposed in Section 3.1 where a set of *ASP*-trees are maintained for streaming spatio-textual objects $\mathcal{D}$.
- **DVs-E**. KMV based algorithm proposed in Section 3.2 where a KMV synopsis is maintained for $\mathcal{D}$.
- **BN-E**. *Local correlation* based algorithm introduced in Section 4 where an A^2SP-tree is continuously maintained for $\mathcal{D}$.

Datasets. The following three real-life datasets are employed in the experiments to evaluate performance of various algorithms. **GN** is obtained from the US board one Geographic Names[4] in which each object is associated with a geographic location and a short text description. **TW** [16] contains 11.5 millions tweets with geo-locations from May 2012 to August 2012, which serves as the default dataset. We also generate dataset **Cars**[26] by obtaining the spatial locations from corresponding spatial datasets from Rtree - Portal[5] and tagging these objects with user-generated textual content from 20 Newsgroups[6]. Table 2 summaries the important statistics of each dataset.

Workload. The workload for selectivity estimation of spatial keyword search consists of 1000 queries, and each query is randomly issued after half of the streaming objects have arrived. The average response time, average relative error and average update time of the algorithms are reported to evaluate the performance. The query region is a rectangle whose center is randomly selected from the locations of the underlying objects. Then we randomly pick m terms from the object as the query keywords, where m is randomly chosen from 1 to 6. Meanwhile, the query region size is randomly chosen from 1% to 10% of the dataset space. The memory budget (B) is measured as the ratio of dataset size, which varies from 0.5% to 5% with default value 2%.

All experiments are carried out on a PC with Intel Xeon 2.40GHz dual CPU and 4G RAM. The operating system is Debian 6.0.4. All algorithms were implemented in C++ and compiled with GCC 4.7.2 with -O3 flag. The threshold ϵ for RC-E is set to 0.00004.

6.2 Performance Evaluation

[4]http://geonames.usgs.gov

[5]http://www.rtreeportal.org

[6]http://people.csail.mit.edu/jrennie/20Newsgroups

6.2.1 Experimental Tuning

Tuning K. We first tune the boosting threshold K for BN-E algorithm by varying the value of K in three datasets under default settings. Figure 5 reports that the performance of BN-E first gets improved with the growth of K, then the relative error starts to rise up for large K values. This is because we cannot properly learn the local BN due to the unstable performance of the KMV synopses when K is small, i.e., there are no sufficient representative samples. On the other hand, we tend to boost $q.R$ to a much larger region for a large K value and hence the *local correlations* w.r.t q may not be well preserved. Given these considerations, the boosting threshold K is set as 75 in the experiments.

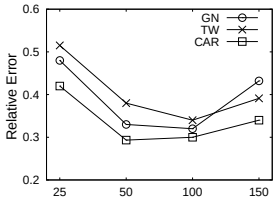


Figure 5: Tuning K

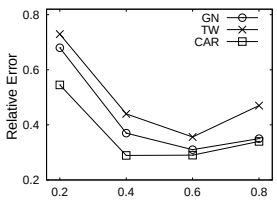
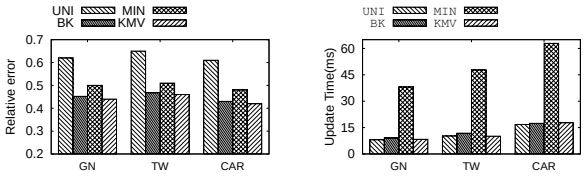


Figure 6: Turning λ

Tuning λ. In BN-E Algorithm, we utilize inwardness of ASP-tree and KMV synopses techniques. A natural concern is the resource allocation between two structures. Figure 6 depicts the impact of resource allocation strategy for A^2SP-tree by reporting the performance of BN-E against different λ values. Recall that for a given memory budget B, the spaces allocated to ASP-tree and KMV synopses are $(1-\lambda)\times B$ and $\lambda \times B$, respectively. As expected, we need a balanced space allocation as both ASP-tree and KMV synopses techniques are essential to the learning of local BN in BN-E Algorithm. We set $\lambda = 0.6$ for all experiments.

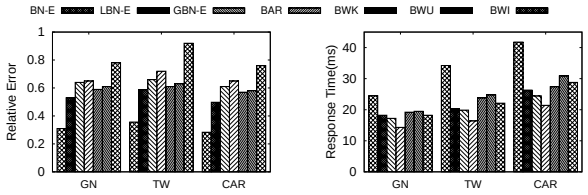


(a) Relative Error (b) Update Time

Figure 7: Compare Different DVs Estimators

Evaluate other DVs estimators. In DVs-E Algorithm, the KMV synopses is a "plug-in" module which can be replaced with any other DVs estimator which supports size estimation on multi-set operations, such as uniform sampling (**UNI**), minhash (**MIN**), and bottom-k (**BK**). Figure 7 compares the performance of these algorithms in terms of relative error and update time. It is interesting that BK achieves almost the same performance compared with KMV synopses, while the performance of MIN is dominated by that of BK and KMV. Note that the update time of MIN is much slower because a couple of hash functions are required by Minhash technique. Due to its simplicity, uniform sampling based algorithm can efficiently support the update of objects. However, compared with other competitors, the estimate quality of UNI is outperformed by a large margin due to its poor performance on size estimation of multi-set operations. Similar trend is observed when we replace the KMV synopses by other DVs estimators in BN-E Algorithm.

Effectiveness of local boosting. To justify the effectiveness of the local boosting approach, we also investigate



(a) Relative Error (b) Response Time

Figure 8: Different Boosting Strategies

several variants of BN-E Algorithm. Firstly, we evaluate the algorithm without local boosting, namely **LNB-E**; that is, directly estimate the selectivity based on the local BN learned w.r.t the query q. Another alternative is to utilize the global correlation, i.e., relying on the correlations among query keywords for all objects, namely **GNB-E**. Following the idea of *local boosting*, we also investigate other alternatives without utilizing the complicated BN structure. Let R denote the boosted area with sufficient samples. We first apply KMV synopses technique to estimate the selectivity of the query g, denoted by $\hat{A}_R$, where $g.\mathcal{V} = q.\mathcal{V}$ and $g.R = R$. Then we estimate the selectivity of q by scaling down $\hat{A}_R$ where $\hat{A} = \rho \times \hat{A}_R$. The scale ratio ρ is calculated based on various boosting strategies including area (**BAR**), single keyword frequency (**BWK**), union size (**BWU**) and intersection size (**BWI**). Specifically, we have $\rho = \frac{Area(q.R)}{Area(R)}$ for BAR, while $\rho = \frac{freq(t_i,q.R)}{freq(t_i,R)}$ for BWK where t_i is a randomly chosen query keyword and $freq(t_i, R)$ is the frequency of the term t_i within the region R. With similar rationale, we derive ρ values for BWU and BWI.

Figure 8(a) compares the estimation accuracy of BN-E with the above algorithms against three real datasets GN, TW and CAR. The experiment demonstrates the superior performances of BN-E against other alternatives. In addition, we have the following important observations.

- The comparison with LNB-E and GNB-E indicates that we need to properly enlarge the search region to accurately learn the local BN, instead of going to two extremes: without boosting or boosting to the global space. This empirically justifies our three key observations in this paper: (i) the correlations among terms exhibit the "locality" property and we cannot resort to global correlations; (ii) insufficient number of KMV samples leads to poor learning of local BN; (iii) the local correlation is well preserved if the search region is properly enlarged.
- According to the comparison against four *scaling approaches*, we stress that it is difficult to capture the local correlation through a simple scaling ratio. On the contrary, the local BN provides an elegant solution.

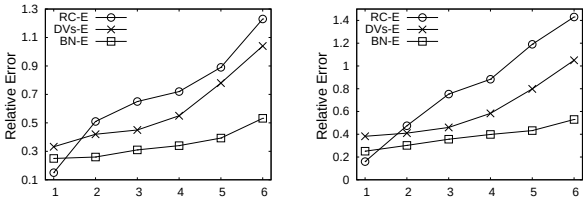
We also evaluate the query response time of these algorithms against three real datasets. Figure 8(b) reports that BN-E has the largest query latency. Nevertheless, considering of the remarkable gain on estimate accuracy which is crucial for selectivity estimation algorithms, BN-E achieves an excellent trade-off between accuracy and query response time.

6.2.2 Estimate Accuracy Evaluation

In this subsection, we assess the effectiveness of three algorithms in term of relative error.

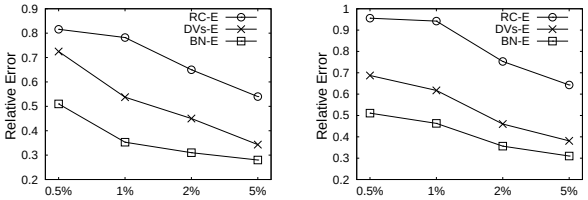
Effect of the number of query keywords (m). In Figure 9, we study the impact of the number of query keywords (m) on two real datasets GN and TW, respectively. As expected, RC-E outperforms other algorithms when there is

only one query keyword, since ASP-tree can provide better estimation in the case of single query keyword query. As shown in Algorithm 3, BN-E becomes RC-E with less space when there is only one query keyword. It is interesting that BN-E still outperforms DVs-E by up to 13%. The performance of RC-E dramatically drops when the number of query keywords grows, which implies the violation of query keywords independence assumption in these real datasets. Although consistently outperforming RC-E for multiple query keywords, the accuracy of DVs-E also degrades significantly when m increases, because of the smaller cardinality of the query and the decreasing number of representative samples. Compared with RC-E and DVs-E, BN-E suffers much less from the growth of m because the local BN learned by BN-E can properly capture the complicated local correlations among query keywords. Figure 9 clearly demonstrates that the advantage of local correlation based computing paradigm becomes more significant when m increases.



(a) GN (b) TW

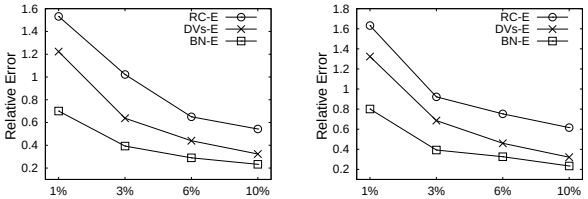
Figure 9: Varying # Query Keywords (m)



(a) GN (b) TW

Figure 10: Varying Space Budget (B)

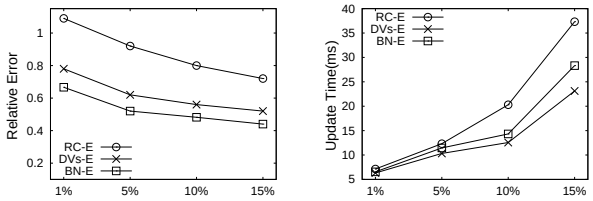
Effect of space budget (B). Figure 10 demonstrates superior performance of BN-E against RC-E and DVs-E by varying the space budget, in term of the percentage of the dataset size. As expected, the performance of all algorithms improves when more memory space is available.



(a) GN (b) TW

Figure 11: Varying Query Region

Effect of Query Region Size. In the last experiment for testing effectiveness, we evaluate the performance of three algorithms against the growth of search region size. As shown in Figure 11, the performance of all algorithms improves against the growth of the region size due to larger result size and greater number of representative samples, and BN-E consistently outperforms RC-E and DVs-E. It is reported that performance gap is more significant when the query region size is small. This demonstrates the effectiveness of our local boosting approach because it will be frequently applied when the search region is small.



(a) Relative Error (b) Update Time

Figure 13: Varying Sliding Window Size

6.2.3 Efficiency Evaluation.

In this subsection, we evaluate the time efficiency of three algorithms in terms of query response time and summary update time. Figure 12(a)-(b) plot the query response time of three algorithms as a function of the space budget on GN and TW datasets, respectively. As expected, the query response time of three algorithms increases due to the larger summary size. Among three algorithms, RC-E has the best performance due to its simplicity.

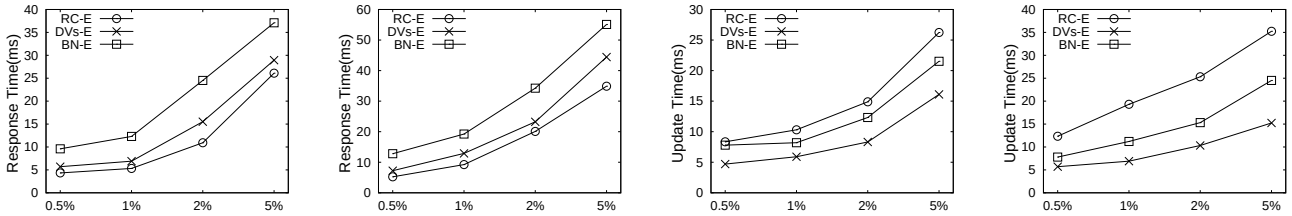
The average update time for each incoming object of the three algorithms are reported in Figure 12(c)-(d). It is shown that the update cost of three algorithms increases with the available space. Among three algorithms, DVs-E achieves the best performance because many objects are discarded after computing their hash values. RC-E has the slowest update time because all objects will be counted by corresponding ASP-trees. As an integration of ASP-tree and KMV synopses, the update cost of A^2SP-tree consistently lies between RC-E and DVs-E under all settings.

6.2.4 Extension Evaluation

Finally, we evaluate the performance of three algorithms on sliding window model as well as boolean spatial keyword queries.

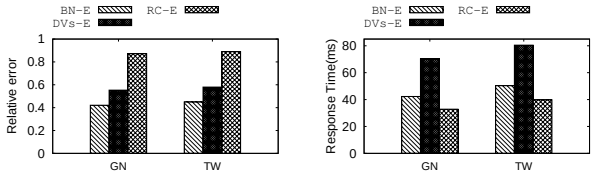
Sliding Window Model. We evaluate the performance of BN-E over sliding window model using tweets dataset with default setting. By varying the size of the sliding window, which is calculated as the percentage of dataset size, we first report the effectiveness of BN-E in Figure 13(a). It is shown that the relative error of all algorithms decreases with the increase of sliding window size due to larger result sizes and representative samples. Figure 13(b) demonstrates that three algorithms can efficiently handle the update of the streams. DVs-E consistently outperforms other two algorithms while the performance of RC-E is always ranked the last.

Boolean Spatial Keyword Query. We evaluate the performance of the boolean spatial query of three algorithms in terms of estimate accuracy and query response time. Particularly, to effectively support NOT operator, we maintain a global ASP-tree in A^2SP-tree for all arriving objects to estimate the number of objects within the search region. Given the queries generated for spatial keyword search, we generate the boolean spatial keyword queries by randomly replacing AND operator by OR operator and adding NOT operator when a query is issued. Figure 14(a) reports the relative error of three algorithms which demonstrates superior performance of BN-E. We also evaluate the query response



(a) GN (b) TW (c) GN (d) TW

Figure 12: Varying Space Budget (B)



(a) Relative Error (b) Response Time

Figure 14: Boolean Query

time of three algorithm in Figure 14(b) where RC-E has the best performance.

7. CONCLUSION

The emerging trend of streaming spatio-textual data opens up a wide variety of novel applications. In this paper, we investigate the problem of selectivity estimation for spatial keyword search which is essential and fundamental for the streaming spatio-textual data analytics. Three algorithms are proposed in this paper. First two are extensions of range counting (RC) estimator and distinct values (DVs) estimator techniques. Observe that the correlation among the terms of the spatio-textual objects exhibits the "locality" property, we develop a novel local correlation based computing paradigm to enhance the accuracy of the estimation, where a localized Bayesian network is constructed on-the-fly for selectivity estimation. Our comprehensive experiments on real-life data empirically verify the effectiveness of the local correlation based approach.

Acknowledgments. Ying Zhang is supported by ARC DE140100679 and DP130103245. Wenjie Zhang is supported by ARC DE120102144 and DP120104168. Xuemin Lin is supported by NSFC61232006, NSFC61021004, ARC DP120104168 and DP110102937. Wei Wang is supported by ARC DP130103401 and DP130103405.

8. REFERENCES

[1] H. Abdelhaq, C. Sengstock, and M. Gertz. Eventweet: Online localized event detection from twitter. *PVLDB*, 6(12):1326–1329, 2013.

[2] Z. Bar-Yossef, T. S. Jayram, R. Kumar, D. Sivakumar, and L. Trevisan. Counting distinct elements in a data stream. In *RANDOM*, pages 1–10, 2002.

[3] K. S. Beyer, P. J. Haas, B. Reinwald, Y. Sismanis, and R. Gemulla. On synopses for distinct-value estimation under multiset operations. In *SIGMOD Conference*, pages 199–210, 2007.

[4] C. Budak, T. Georgiou, D. Agrawal, and A. El Abbadi. Geoscope: Online detection of geo-correlated information trends in social networks. *PVLDB*, 7(4), 2013.

[5] L. Chen, G. Cong, and X. Cao. An efficient query indexing mechanism for filtering geo-textual data. In *SIGMOD Conference*, 2013.

[6] Y.-Y. Chen, T. Suel, and A. Markowetz. Efficient query processing in geographic web search engines. In *SIGMOD Conference*, 2006.

[7] Z. Chen, F. Korn, N. Koudas, and S. Muthukrishnan. Selectivity estimation for boolean queries. In *PODS*, pages 216–225, 2000.

[8] C. K. Chow and C. N. Liu. Approximating discrete probability distributions with dependence trees. *IEEE Transactions on Information Theory*, 14(3), 1968.

[9] E. Cohen, G. Cormode, and N. G. Duffield. Structure-aware sampling on data streams. In *SIGMETRICS*, pages 197–208, 2011.

[10] E. Cohen and H. Kaplan. Tighter estimation using bottom k sketches. *PVLDB*, 1(1), 2008.

[11] A. Das, J. Gehrke, and M. Riedewald. Approximation techniques for spatial data. In *SIGMOD Conference*, 2004.

[12] M. Eftekhar and N. Koudas. Some research opportunities on twitter advertising. *IEEE Data Eng. Bull.*, 36(3), 2013.

[13] M. Hadjieleftheriou, X. Yu, N. Koudas, and D. Srivastava. Hashed samples: selectivity estimators for set similarity selection queries. *PVLDB*, 1(1), 2008.

[14] J. Hershberger, N. Shrivastava, S. Suri, and C. D. Tóth. Adaptive spatial partitioning for multidimensional data streams. *Algorithmica*, 46(1), 2006.

[15] R. Kaushik and D. Suciu. Consistent histograms in the presence of distinct value counts. *PVLDB*, 2(1), 2009.

[16] G. Li, Y. Wang, T. Wang, and J. Feng. Location-aware publish/subscribe. In *KDD*, 2013.

[17] C. Long, R. C.-W. Wong, K. Wang, and A. W.-C. Fu. Collective spatial keyword queries: a distance owner-driven approach. In *SIGMOD*, 2013.

[18] R. Pagh, M. Stckel, and D. P. Woodruff. Is min-wise hashing optimal for summarizing set intersection? In *PODS*, 2014.

[19] D. Papadias, Y. Tao, G. Fu, and B. Seeger. Progressive skyline computation in database systems. *ACM Trans. Database Syst.*, 30(1), 2005.

[20] V. Poosala, Y. E. Ioannidis, P. J. Haas, and E. J. Shekita. Improved histograms for selectivity estimation of range predicates. In *SIGMOD Conference*, 1996.

[21] A. Skovsgaard, D. Sidlauskas, and C. S. Jensen. Scalable top-k spatio-temporal term querying. In *ICDE*, 2014.

[22] N. Thaper, S. Guha, P. Indyk, and N. Koudas. Dynamic multidimensional histograms. In *SIGMOD Conference*, pages 428–439, 2002.

[23] K. Tzoumas, A. Deshpande, and C. S. Jensen. Efficiently adapting graphical models for selectivity estimation. *VLDB J.*, 22(1), 2013.

[24] S. Vaid, C. B. Jones, H. Joho, and M. Sanderson. Spatio-textual indexing for geographical search on the web. In *SSTD*, 2005.

[25] D. Wu, G. Cong, and C. S. Jensen. A framework for efficient spatial web object retrieval. *VLDB J.*, 2012.

[26] C. Zhang, Y. Zhang, W. Zhang, and X. Lin. Inverted linear quadtree: Efficient top k spatial keyword search. In *ICDE*, 2013.

[27] C. Zimmer, C. Tryfonopoulos, and G. Weikum. Exploiting correlated keywords to improve approximate information filtering. In *SIGIR*, 2008.