Star-Join: Spatio-Textual Similarity Join

Sitong Liu Guoliang Li Jianhua Feng
Department of Computer Science and Technology, Tsinghua University, Beijing 100084, China.
liu-st10@mails.tsinghua.edu.cn, liguoliang@tsinghua.edu.cn, fengjh@tsinghua.edu.cn

ABSTRACT

Location-based services have attracted significant attention due to modern mobile phones equipped with GPS devices. These services generate large amounts of spatio-textual data which contain both spatial location and textual descriptions. Since a spatio-textual object may have different representations, possibly because of deviations of GPS or different user descriptions, it calls for efficient methods to integrate spatio-textual data from different sources. In this paper we study a new research problem called spatio-textual similarity join: given two sets of spatio-textual objects, we find the similar object pairs. To the best of our knowledge, we are the first to study this problem. We make the following contributions: (1) We develop a filter-and-refine framework and devise several efficient algorithms. We first generate spatial and textual signatures for the objects and build inverted index on top of these signatures. Then we generate candidate pairs using the inverted lists of signatures. Finally we refine the candidates and generate the final result. (2) We study how to generate high-quality signatures for spatial information. We develop an MBR-prefix based signature to prune large numbers of dissimilar object pairs. (3) Experimental results on real and synthetic datasets show that our algorithms achieve high performance and scale well.

Categories and Subject Descriptors

H.2.8 [**Database Applications**]: Spatial databases and GIS; H.3.3 [**Information Search and Retrieval**]

General Terms

Algorithms, Experimentation, Performance

Keywords

Spatio-Textual, Similarity Join, MBR-Prefix

1. INTRODUCTION

With the near ubiquity of global position systems (GPS) in smartphones, location-based services (LBS) have recently attracted significant attentions from both academic and industrial community. These services generate large amounts of spatio-textual data which contain both geographical location and textual description. As a spatio-textual object may have different representations, possibly because of deviations of GPS or different user descriptions, it calls for efficient methods to correlate the spatio-textual data from different sources. For example, Google Map[1] generates the detailed information of points of interests (e.g., hotels and restaurants) by integrating the relevant data from multiple sources. Factual[2] extracts spatio-textual information from the user-generated data to generate new points of interest.

In this paper, we study a new research problem, called Spatio-textual similarity join (`StarJoin`). Given two sets of spatio-textual objects with a spatial region and textual descriptions, it finds all similar object pairs. Two objects are similar if their spatial similarity and textual similarity are larger than given thresholds. In this paper, we use Jaccard coefficient as an example to quantify the spatial similarity and textual similarity and our techniques can be easily extended to support other similarity functions. `StarJoin` has many real applications. One example is Tuan800[3], a famous information integration service which integrates discount information from various group-on websites. Each discount message is associated with a spatial range and some keyword descriptions. Since different group-on websites may contain many similar discount messages, it is very important to perform a similarity join on the datasets so as to eliminate redundant ones for improving user experiences.

There are some recent studies on spatial join [12, 9, 13, 3] and string similarity join [4, 1]. Although we can extend their methods to support our problem, they are rather inefficient as they only use spatial pruning or textual pruning, and may generate large numbers of intermediate results. To address this limitation and improve the performance, we develop a filter-and-refine framework. First we generate spatial and textual signatures for the objects and build inverted indexes to avoid redundant computations. Next we use the signatures to find candidate pairs whose signatures are similar enough. Finally we verify the candidates to get the final answers. We propose several algorithms by organizing spatial and textual signatures in different ways. In addition, we propose an MBR-Prefix filter technique to generate high-quality signatures. For each object, it selects a subregion of the object as a spatial signature to substitute the entire

CIKM'12, October 29–November 2, 2012, Maui, HI, USA.

[1]http://maps.google.com
[2]http://www.factual.com
[3]http://www.tuan800.com

region. We also prove that the selected subregion is minimized. To summarize, we make the following contributions:
(1) We study a new research problem called Spatio-textual Similarity Join. We explore a filter-and-refine framework and propose efficient algorithms which can prune large numbers of dissimilar objects.
(2) We develop an MBR-Prefix based signature which uses subregions of objects as signatures to support spatial pruning. We prove that the selected subregion is minimized.
(3) We have conducted extensive experiments on real and synthetic datasets. Experimental results show that our methods achieve high performance and scale well.

2. PRELIMINARIES

We first formulate the problem of spatio-textual similarity join in Section 2.1, and then introduce prefix filter property in Section 2.2.

2.1 Problem Statement

Consider two collections of objects $\mathcal{R} = \{r_1, r_2..., r_n\}$ and $\mathcal{S} = \{s_1, s_2..., s_m\}$. Each object r (or s) includes a spatial region $\mathcal{M}_r$ and textual description $\mathcal{T}_r$. In this paper we use Minimum Bounding Rectangle (MBR) to capture the spatial information, denoted by $\mathcal{M}_r = [r_{bl}, r_{tr}]$, where $r_{bl} = (r_{bl}.x, r_{bl}.y)$ is the bottom-left point and $r_{tr} = (r_{tr}.x, r_{tr}.y)$ is the top-right point. We use a set of tokens to capture the textual description, denoted by $\mathcal{T}_r = \{t_1, t_2, \ldots, t_v\}$, which describes an object (e.g., {`Hotel, Pizza`}) or users' interests (e.g., {`Seaside, Delivery`}). As tokens may have different importance, we assign each token t_i with a weight $w(t_i)$ (e.g., inverse document frequency `idf`).

To quantify the similarity between two objects, we use the well-known Jaccard as an example to evaluate the spatial similarity (S_{Jac}) and textual similarity (T_{Jac}). Our techniques can be easily extended to support other functions. Due to space constraints, we do not discuss the details.

Definition 1 (Spatial Jaccard). *Given two objects r and s, their spatial Jaccard similarity (S_{Jac}) is defined as:*

$$S_{Jac}(r, s) = \frac{|\mathcal{M}_r \cap \mathcal{M}_s|}{|\mathcal{M}_r| + |\mathcal{M}_s| - |\mathcal{M}_r \cap \mathcal{M}_s|}$$

where $|\cdot|$ is the size of an MBR.

Definition 2 (Textual Jaccard). *Given two objects r and s, their textual Jaccard similarity (T_{Jac}) is defined as:*

$$T_{Jac}(r, s) = \frac{\sum_{t \in \mathcal{T}_r \cap \mathcal{T}_s} w(t)}{\sum_{t \in \mathcal{T}_r \cup \mathcal{T}_s} w(t)}$$

where $w(t)$ is the weight of token t.

Two objects r and s are similar if they satisfy (1) Spatial constraint: their spatial Jaccard similarity is larger than a spatial similarity threshold τ_s, i.e., $S_{Jac}(r, s) > \tau_s$; and (2) Textual constraint: their textual Jaccard similarity is larger than a textual similarity threshold τ_t, i.e., $T_{Jac}(r, s) > \tau_t$. We formulate the spatio-textual similarity join problem.

Definition 3 (Spatio-Textual Similarity Join). *Given two collections of objects $\mathcal{R} = \{r_1, r_2..., r_n\}$, $\mathcal{S} = \{s_1, s_2..., s_m\}$, and two similarity thresholds τ_s and τ_t, a spatial-textual similarity join finds all similar pairs (r_i, s_j) where $S_{Jac}(r_i, s_j) > \tau_s$ and $T_{Jac}(r_i, s_j) > \tau_t$.*

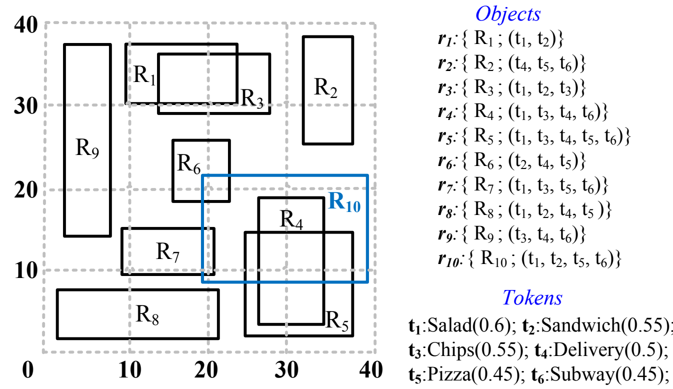


Figure 1: An example of spatio-textual objects

Example 1. Consider the ten objects in Figure 1. Suppose $\tau_s = 0.6$ and $\tau_t = 0.7$. For object pair (r_1,r_3) where $R_1 = [(10, 30), (24, 37)]$ and $R_3 = [(11, 29), (26, 36)]$, $S_{Jac}(r_1, r_3) = \frac{13\times6}{14\times7+15\times7-13\times6} = 0.624$ and $T_{Jac}(r_1, r_3) = \frac{0.6+0.55}{0.6+0.55+0.55} = 0.676$. Thus, r_1 and r_3 are not similar since $0.676 < \tau_t$.

Discussions: For ease of presentation, we suppose $\mathcal{R} = \mathcal{S}$ and our techniques can be easily extended to the case $\mathcal{R} \neq \mathcal{S}$.

2.2 Prefix Filter Property

Prefix filter [4] is a widely used method for solving string similarity join problem. Its basic idea is that if the similarity of two token sets is larger than a given threshold, they should share enough common tokens. Consider two token sets $\mathcal{T}_r$ and $\mathcal{T}_s$. When $w(t_i) = 1$, according to [4], if $T_{Jac}(\mathcal{T}_r, \mathcal{T}_s) > \tau_t$, then $|\mathcal{T}_r \cap \mathcal{T}_s| > \tau_t \times |\mathcal{T}_r| \geq \lfloor \tau_t \times |\mathcal{T}_r| \rfloor + 1$. Based on this property, we first sort all the tokens according to a global order, e.g., `idf`. For each token set $\mathcal{T}$, we generate its prefix by deleting last $\lfloor \tau_t \cdot |\mathcal{T}| \rfloor$ tokens. If any two objects are similar, they must have common tokens in their prefixes. We extend it to the case $w(t_i) \neq 1$. We first sort objects in the descending order of weights. Since $|\mathcal{T}_r \cap \mathcal{T}_s| > \tau_t \times \sum_{i=1}^{|\mathcal{T}_r|} w(t_i)$, then we generate prefix by deleting the last k tokens which satisfy $\sum_{i=|\mathcal{T}|-k+1}^{|\mathcal{T}|} w(t_i) \leq \tau_t \cdot \sum_{i=1}^{|\mathcal{T}|} w(t_i)$ and $\sum_{i=|\mathcal{T}|-k}^{|\mathcal{T}|} w(t_i) > \tau_t \cdot \sum_{i=1}^{|\mathcal{T}|} w(t_i)$. All the objects containing common tokens in their prefixes will be taken as candidates.

3. PREFIX FILTER BASED METHODS

In this section, we propose a filter-and-refine framework (Section 3.1) and devise five filtering algorithms (Section 3.2).

3.1 A Filter-and-Refine Framework

To avoid enumerating every object pair, we introduce an incremental signature-based framework. We scan the objects in order and maintain index for all the objects that have been visited. The framework includes three steps:
Filter: For the current object r, we generate its signature $\textsc{Sig}(r)$ and use it to probe the inverted index for candidates. In this paper, the signatures should satisfy the following property. If objects r and s are similar, $\textsc{Sig}(r) \cap \textsc{Sig}(s) \neq \phi$.
Index Update: After finding all the candidates of object r, we insert $\textsc{Sig}(r)$ to the current index.
Refine: We refine all the candidate pairs and check whether they satisfy spatial and textual constraints simultaneous.

3.2 Generating Spatial Prefixes

We now discuss how to generate spatial and textual signatures and how to organize these signatures.

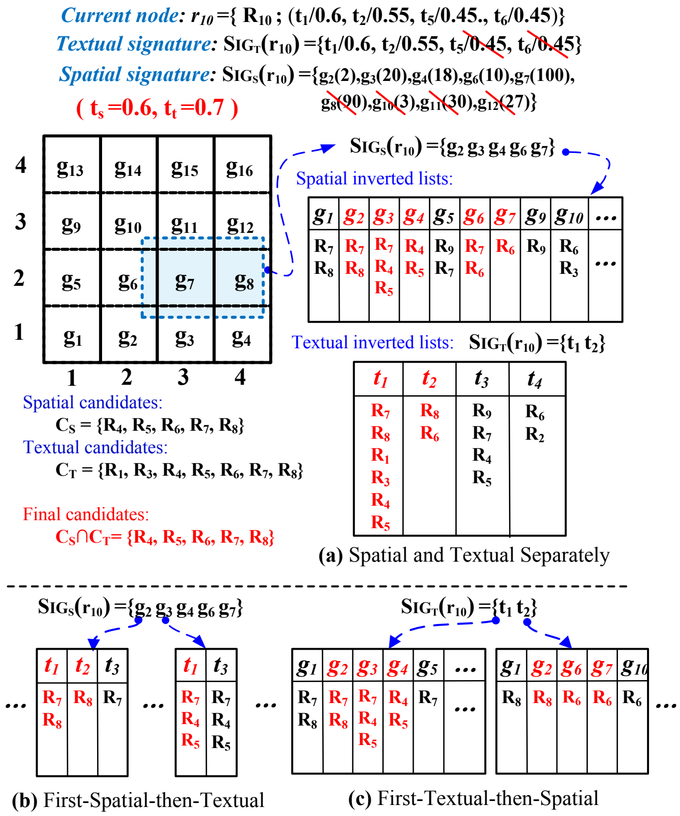


Figure 2: Three prefix-filter based solutions

Signature Generation: Given an object r, we use its token prefix $\mathcal{T}_r$ as its textual signature (Section 2.2), denoted by $\textsc{Sig}_T(r)$. To estimate the region of an object, we partition the space into grids and associate each object r with the grids it intersects, denoted as $\mathcal{G}_r$. Similar to textual prefix filter property, we can generate spatial signatures as follows. Given an object r, we first sort grids in $\mathcal{G}_r$ based on a global order. Then we delete last k grids which satisfy $\sum_{i=|\mathcal{G}_r|-k+1}^{|\mathcal{G}_r|} |\mathcal{M}_{g_i} \cap \mathcal{M}_r| \le \tau_s \cdot |\mathcal{M}_r|$ and $\sum_{i=|\mathcal{G}_r|-k}^{|\mathcal{G}_r|} |\mathcal{M}_{g_i} \cap \mathcal{M}_r| > \tau_s \cdot |\mathcal{M}_r|$, denoted by $\textsc{Sig}_S(r)$. If r and s are similar, they at least share one common grid in their spatial signatures as stated in Lemma 1.

LEMMA 1. *Given two objects r and s, $\textsc{Sig}_S(r)$ and $\textsc{Sig}_S(s)$ are spatial signatures of r and s. If $S_{Jac}(r,s) > \tau_s$, then $\textsc{Sig}_S(r) \cap \textsc{Sig}_S(s) \neq \phi$.*

Signature Organization: We study how to organize these signatures and discuss six possible solutions.

(1) Spatial Only: We only use the spatial signatures to build inverted index. Each entry in the index is a grid which maintains a list of objects overlapping with the grid.

(2) Textual Only: We only use the textual signatures to build inverted index. Each entry in the index is a token which keeps a list of objects containing the token. Obviously, these two methods are ineffective since they only use a single constraint to prune candidates.

(3) Spatial and Textual Separately: We build inverted index for spatial and textual signatures separately. The algorithm is illustrated in Figure 2(a). For the current object r, we generate its spatial signature $\textsc{Sig}_S(r)$ and textual signature $\textsc{Sig}_T(r)$. For each grid $g \in \textsc{Sig}_S(r)$, we use it to probe the corresponding spatial inverted index and add all the objects in these lists to spatial candidate set $\mathcal{C}_S$. Meanwhile, for each token $t \in \textsc{Sig}_T(r)$, we scan the corresponding textual inverted list and add the objects to textual candidate set $\mathcal{C}_T$. Then the intersection of $\mathcal{C}_S$ and $\mathcal{C}_T$ will be taken as the candidates. Notice that this method is not efficient since the process of spatial and textual filtering is independent.

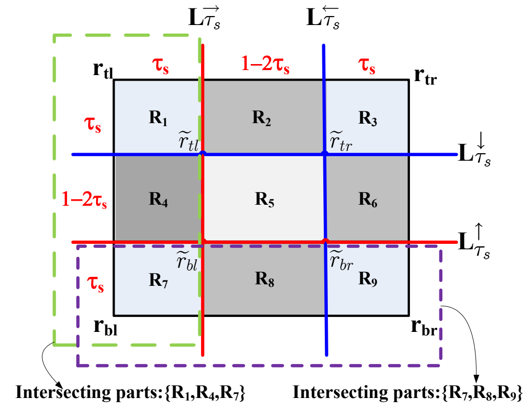


Figure 3: MBR-Prefix filtering technique: $\tau_s \le 0.5$

(4) First-spatial-then-textual: We can build a two-layer index. If the spatial index is arranged at the top layer, the index is called first-spatial-then-textual (Figure 2(b)). Given an object r, we first generate its spatial signature $\textsc{Sig}_S(r)$ and textual signature $\textsc{Sig}_T(r)$. The top layer is the grids in $\textsc{Sig}_S(r)$ and the bottom layer is the inverted lists for tokens in $\textsc{Sig}_T(r)$. To find the candidates of r, for each grid g in $\textsc{Sig}_S(r)$ and each token t in $\textsc{Sig}_T(r)$, if t is in the inverted index of g, the objects in the corresponding inverted lists are candidates. Similarly we can update the index for object r.

(5) First-textual-then-spatial: If the textual index is arranged at the top layer, the index is called first-textual-then-spatial. In Figure 2(c), we illustrate the algorithm. Given an object r, we first generate its spatial signature $\textsc{Sig}_S(r)$ and textual signatures $\textsc{Sig}_T(r)$. For each token t in $\textsc{Sig}_T(r)$ and grid g in $\textsc{Sig}_S(r)$, if g is in the inverted index of token t, the objects in the corresponding inverted list are candidates. Similarly we can update the index for object r.

4. MBR-PREFIX BASED FILTERING

The grid based spatial filter has a limitation that the filtering power relies largely on grid granularity. To address this problem, we propose an MBR-Prefix based filtering technique which uses more accurate spatial information to generate signatures. According to prefix filter property, two objects are similar in space only if they have enough overlap. Thus, we only need to keep specific subregion of an object. To illustrate the idea clearly, we first introduce some concepts: MBR-Prefix, Representative MBR-Prefix and Minimum MBR-Prefix.

DEFINITION 4. (MBR-PREFIX, REPRESENTATIVE MBR-PREFIX *and* MINIMUM MBR-PREFIX) *Given an object r, any subregion of $\mathcal{M}_r$ is called an* MBR-Prefix *of r. An MBR-Prefix $\mathcal{M}_p$ is called a* representative MBR-Prefix *of r, if for any object s which satisfies $S_{Jac}(r,s) > \tau_s$, we have $\mathcal{M}_p \cap \mathcal{M}_s \neq \phi$. The representative MBR-Prefix with the minimum size is called the* Minimum MBR-Prefix.

Now we discuss how to generate the minimum MBR-Prefix in three cases: $\tau_s < 0.5$, $\tau_s = 0.5$ and $\tau_s > 0.5$.

Case 1 - $\tau_s < 0.5$: Consider the MBR of object r, denoted by $\mathcal{M}(r_{bl}, r_{tr})$, in Figure 3. Its projection along $x(y)$ axis is

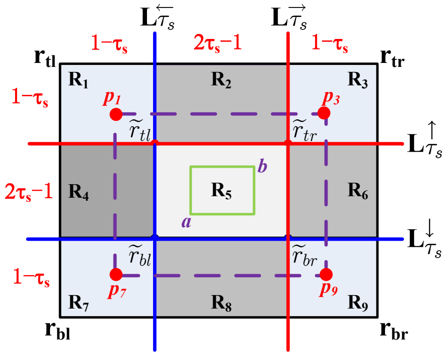


Figure 4: MBR-Prefix filtering technique: $\tau_s > 0.5$

denoted by $width(height)$. Let Line(r_{bl}, r_{tl}) denote the left line of the MBR, i.e., the line from the bottom-left point to the top-left point. Let $L^{\rightarrow}_{\tau_s}$ denote the parallel line of the left line with distance $\tau_s \times width$. Similarly we can define the right line, the bottom line, and the top line. Let $L^{\leftarrow}_{\tau_s}$ denote the parallel line of the right line with distance $\tau_s \times width$. Let $L^{\downarrow}_{\tau_s}$ and $L^{\uparrow}_{\tau_s}$ respectively denote the parallel lines of the top line and the bottom line with distance $\tau_s \times height$.

These four lines generate four intersections $\widetilde{r}_{bl}$, $\widetilde{r}_{tl}$, $\widetilde{r}_{tr}$, $\widetilde{r}_{br}$, and divide $\mathcal{M}_r$ into nine regions ($R_1 \sim R_9$) as illustrated in Figure 3 where we omit $width$ and $height$ in the figure for concise illustration. Notice that the size of the region on the left of $L^{\rightarrow}_{\tau_s}$ is $\tau_s \times |\mathcal{M}_r|$, and we use $\mathcal{M}_{(R_1 \cup R_4 \cup R_7)}$ to denote this region. For any object s which is similar to r, we have $|\mathcal{M}_s \cap \mathcal{M}_r| > \tau_s \times |\mathcal{M}_r|$. Thus, there must be at least one point of s falling into the area on the right of Line $L^{\rightarrow}_{\tau_s}$, *i.e.*, $\mathcal{M}_r \setminus \mathcal{M}_{(R_1 \cup R_4 \cup R_7)}$ (otherwise the intersecting part of r and s can not be larger than $\tau_s \times |\mathcal{M}_r|$). Thus, $\mathcal{M}_r \setminus \mathcal{M}_{(R_1 \cup R_4 \cup R_7)}$ is a representative MBR-Prefix of r. Similarly, $\mathcal{M}_r \setminus \mathcal{M}_{(R_1 \cup R_2 \cup R_3)}$, $\mathcal{M}_r \setminus \mathcal{M}_{(R_3 \cup R_6 \cup R_9)}$ and $\mathcal{M}_r \setminus \mathcal{M}_{(R_7 \cup R_8 \cup R_9)}$ are all representative MBR-Prefixes. We now prove that their intersection area, i.e., $\mathcal{M}_{R_5}$, is the minimum MBR-Prefix of r.

LEMMA 2. *Given an object r, if $\tau_s < 0.5$ then $\mathcal{M}_{R_5}$ is the minimum MBR-Prefix of r.*

Recall the algorithms in Section 3.2, for each object r, we use all the grids which have overlap with the entire $\mathcal{M}_r$ as its spatial signature. According to Lemma 2, only those objects intersecting with $\mathcal{M}_{R_5}$ can be similar to r. Then we only need to keep grids intersecting with $\mathcal{M}_{R_5}$ as spatial signature, denoted as $\mathcal{G}^p_r$. Take the separated algorithm as an example. When coming an object r, for any object s that has been visited, if r is similar to s, r must have overlap with at least one grid in $\mathcal{G}^p_s$, i.e., $\mathcal{G}_r \cap \mathcal{G}^p_s \neq \phi$. Thus we can take the objects in the inverted list of each grid in $\mathcal{G}_r$ as candidates in terms of spatial constraints. After finding all the candidates of r, we update the index by inserting its MBR-Prefix into the inverted lists of $\mathcal{G}^p_r$.

Case 2 - $\tau_s = 0.5$: As shown in Figure 3, line $L^{\rightarrow}_{\tau_s}$ and line $L^{\leftarrow}_{\tau_s}$ coincide with each other, and so do line $L^{\downarrow}_{\tau_s}$ and line $L^{\uparrow}_{\tau_s}$. Thus, $\mathcal{M}_{R_5}$ turns into a point (denoted by p). Similar to the former case, we can use point p as its minimum MBR-Prefix to represent the entire MBR. All the objects without intersection with p can be pruned.

Case 3 - $\tau_s > 0.5$: Notice that when τ_s increases, line $L^{\rightarrow}_{\tau_s}$ and line $L^{\leftarrow}_{\tau_s}$ moves towards each other and $\mathcal{M}_{R_5}$ becomes smaller. Especially, when $\tau_s > 0.5$, line $L^{\rightarrow}_{\tau_s}$ moves to the right side of line $L^{\leftarrow}_{\tau_s}$ as shown in Figure 4. Like the former case, we can also use the center of R_5 to represent the whole area and all the objects covering this point will be taken as candidates. However, this bound is not tight. For example, consider the MBR-Prefix $\mathcal{M}_{R_2 \cup R_3 \cup R_5 \cup R_6}$. Though it covers point p, it cannot be a candidate since $|\mathcal{M}_{R_2 \cup R_3 \cup R_5 \cup R_6}| = \tau_s^2 \cdot |\mathcal{M}_r| \le \tau_s \cdot |\mathcal{M}_r|$. To this end, we propose new techniques for $\tau_s > 0.5$.

Table 1: Dataset statistics.

Property	CN	Twitter
Object number (Million)	10	10
Data size (GB)	0.817	0.801
Index size (GB)	1.015	1.14

First if $\tau_s > 0.5$ then $\mathcal{M}_{R_1}$, $\mathcal{M}_{R_3}$, $\mathcal{M}_{R_7}$ and $\mathcal{M}_{R_9}$ are representative MBR-Prefixes of r as formalized in Lemma 3.

LEMMA 3. *Given an object r, if $\tau_s > 0.5$, $\mathcal{M}_{R_1}$, $\mathcal{M}_{R_3}$, $\mathcal{M}_{R_7}$ and $\mathcal{M}_{R_9}$ are representative MBR-Prefixes of r.*

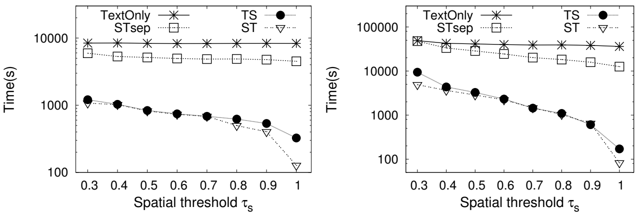
According to Lemma 3, s must have overlap with regions R_1, R_3, R_7 and R_9 of r simultaneously. Suppose p_1,p_3,p_7,p_9 are four points of s falling in R_1, R_3, R_7 and R_9 as shown in Figure 4. Obviously, $\mathcal{M}(p_7, p_3) \subseteq \mathcal{M}_s$ since p_7 and p_3 are two inner points of s. Thus, *we have an intuition that $\mathcal{M}_s$ must cover some subregions of r*. We can use this property to improve the MBR-Prefix. To illustrate our idea more clearly, we introduce some new concepts, called Coverage MBR-Prefix and Maximum-Coverage MBR-Prefix.

DEFINITION 5. (COVERAGE MBR-PREFIX AND MAXIMUM-COVERAGE MBR-PREFIX) *Given an object r, an MBR-Prefix $\mathcal{M}_p$ of r is called a Coverage MBR-Prefix of r, if for any object s satisfying $S_{Jac}(r, s) > \tau_s$, we have $\mathcal{M}_p \subseteq \mathcal{M}_s$. Among all these Coverage MBR-Prefixes, we call the largest one as the Maximum-Coverage MBR-Prefix.*

We now prove that if $\tau_s > 0.5$, $\mathcal{M}_{R_5}$ is the Maximum-Coverage MBR-Prefix of r as stated in Lemma 4.

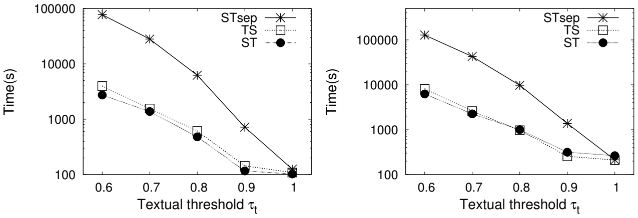
LEMMA 4. *Given an object r, if $\tau_s > 0.5$, $\mathcal{M}_{R_5}$ is the Maximum-Coverage MBR-Prefix of r.*

According to Lemma 4, only the objects entirely covering $\mathcal{M}_{R_5}$ can be candidates. That is, the pivotal points $\widetilde{r}_{tl}, \widetilde{r}_{tr}, \widetilde{r}_{bl}, \widetilde{r}_{br}$ should be covered simultaneously. Notice that these four points are actually determined by two x-coordinates and two y-coordinates since adjacent points have the same x-coordinate or y-coordinate. If we utilize the order of coordinates while building index, then only two points are needed for locating $\mathcal{M}_{R_5}$. Suppose the objects are sorted according to x-coordinate of the right line previously. For each object r, we only need to keep the grids which intersect with Line$(\widetilde{r}_{bl}, \widetilde{r}_{tl})$ as its spatial signature. Take the separated index based method as an example. When coming an object r, for any object s that have been visited, according to the analysis above, if r is similar to s, r must cover the signature of s, that is $\mathcal{G}_r$ contains $\mathcal{G}^p_s$. Thus we can take the objects in the inverted list of each grid in $\mathcal{G}_r$ as candidates in terms of spatial constraints. Notice that we do not need to scan the entire inverted list. Based on the definition of maximum coverage MBR-prefix, r must totally cover $\mathcal{M}_{r_5}$ of s, that is, $\widetilde{r}_{tr}.x < s_{tr}.x \le r_{tr}.x$. Thus, for each inverted list, we only need to scan the objects between Line$(\widetilde{r}_{br}, \widetilde{r}_{tr})$ and Line(r_{br}, r_{tr}). After finding all the candidates of r, we need insert its MBR-Prefix into the inverted lists of $\mathcal{G}^p_r$.



(a) CN (b) Twitter

Figure 5: Evaluating the effect of different τ_s



(a) CN (b) Twitter

Figure 6: Evaluating the effect of different τ_t

5. EXPERIMENTAL STUDY

We conducted extensive experiments on two datasets to evaluate our proposed techniques.

5.1 Experimental Settings

Datasets and Experimental Environment: We use two datasets: CN and Twitter (Table 1). Twitter is a real dataset. We crawled 10 million tweets with region and textual information from Twitter. CN is a synthetic dataset which combines the MBRs of China and publications in DBLP randomly. All the algorithms were implemented in C++ and run on a Linux machine with an Intel(R) Xeon(R) CPU X5670 @ 2.93GHz and 48GB memory.

5.2 Evaluating Different Signature Schemes

We evaluate the five signature schemes, spatial only (`SpaOnly`), textual only (`TextOnly`), spatial and textual separate (`STsep`), first-spatial-then-textual (`ST`), first-textual-then-spatial (`TS`), in Section 3 by varying τ_s and τ_t. Figures 5 and Figure 6 show the results. Since `SpaOnly` was much slower than other methods, we omitted it in the figures. `ST` and `TS` almost had the same performance and outperformed significantly than other methods.

5.3 MBR-Prefix vs Non-MBR-Prefix

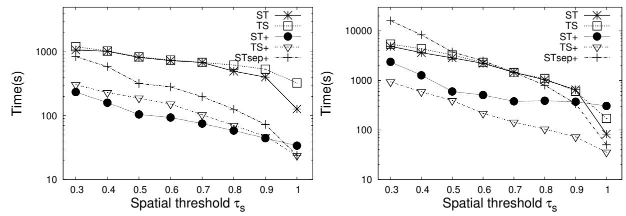
We evaluate MBR-Prefix based filtering techniques. Figures 7 and 8 show the results. The algorithms with + denote the improved algorithm by incorporating MBR-Prefix. We can observe that the MBR-Prefix technique significantly improved the performance of original algorithms.

6. RELATED WORK

Spatial Join: Many methods have been proposed to study the spatial join problem [3, 12, 8]. [3] used R-tree like structure [7] to organize spatial data. [12] used hash methods by partitioning the space into grids. [8] gave a survey about existing spatial join techniques.

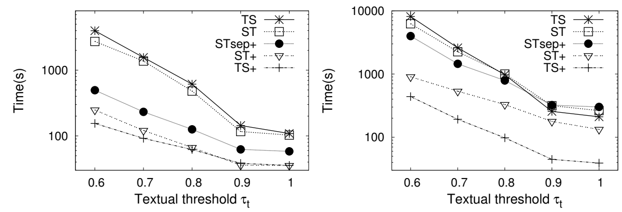
String Similarity Join: Recently there are many studies on string similarity joins [2, 14, 10].

Spatial Keyword Search: There are many studies on spatial keyword search [6, 5, 11] which integrated inverted indexes and R-tree to support spatial keyword search.



(a) CN (b) Twitter

Figure 7: MBR-Prefix vs Non-MBR-Prefix (τ_s)



(a) CN (b) Twitter

Figure 8: MBR-Prefix vs Non-MBR-Prefix (τ_t)

7. CONCLUSION

In this paper, we study a new research problem called spatio-textual similarity join. We devise a filter-and-refine framework and propose several possible solutions. We further develop an MBR-Prefix based filtering techniqueExperiments show that our methods achieve high performance.

8. ACKNOWLEDGEMENT

This work was partly supported by the National Natural Science Foundation of China under Grant No. 61003004, the National Grand Fundamental Research 973 Program of China under Grant No. 2011CB302206, a project of Tsinghua University under Grant No. 20111081073, and the "NExT Research Center" funded by MDA, Singapore, under the Grant No. WBS:R-252-300-001-490.

9. REFERENCES

[1] A. Arasu, V. Ganti, and R. Kaushik. Efficient exact set-similarity joins. In *VLDB*, pages 918–929, 2006.

[2] R. J. Bayardo, Y. Ma, and R. Srikant. Scaling up all pairs similarity search. In *WWW*, pages 131–140, 2007.

[3] T. Brinkhoff, H.-P. Kriegel, and B. Seeger. Efficient processing of spatial joins using r-trees. In *SIGMOD Conference*, pages 237–246, 1993.

[4] S. Chaudhuri, V. Ganti, and R. Kaushik. A primitive operator for similarity joins in data cleaning. In *ICDE*, page 5, 2006.

[5] J. Fan, G. Li, L. Zhou, S. Chen, and J. hu. Seal: Spatio-textual similarity search. *PVLDB*, 2(1):337–348, 2012.

[6] I. D. Felipe, V. Hristidis, and N. Rishe. Keyword search on spatial databases. In *ICDE*, 2008.

[7] A. Guttman. R-trees: A dynamic index structure for spatial searching. In *SIGMOD Conference*, pages 47–57, 1984.

[8] E. H. Jacox and H. Samet. Spatial join techniques. *ACM Trans. Database Syst.*, 32(1):7, 2007.

[9] N. Koudas and K. C. Sevcik. Size separation spatial join. In *SIGMOD Conference*, pages 324–335, 1997.

[10] G. Li, D. Deng, J. Wang, and J. Feng. Pass-join: A partition-based method for similarity joins. *PVLDB*, 5(3):253–264, 2011.

[11] G. Li, J. Feng, and J. Xu. Desks: Direction-aware spatial keyword search. In *ICDE*, pages 474–485, 2012.

[12] M.-L. Lo and C. V. Ravishankar. Spatial hash-joins. In *SIGMOD Conference*, pages 247–258, 1996.

[13] J. M. Patel and D. J. DeWitt. Partition based spatial-merge join. In *SIGMOD Conference*, pages 259–270, 1996.

[14] C. Xiao, W. Wang, and X. Lin. Ed-join: an efficient algorithm for similarity joins with edit distance constraints. *PVLDB*, 1(1):933–944, 2008.