Scaling up cosine interesting pattern discovery: A depth-first method

Jie Cao, Zhiang Wu, Junjie Wu

Jiangsu Provincial Key Laboratory of E-Business, Nanjing University of Finance and Economics, Nanjing 210003, China
Beijing Key Laboratory of Emergency Support Simulation Technologies for City Operations, School of Economics and Management, Beihang University, Beijing 100191, China

Abstract

This paper presents an efficient algorithm called CosMiner, for interesting pattern discovery. The widely used cosine similarity, found to possess the null-invariance property and the anti-cross-support-pattern property, is adopted as the interestingness measure in CosMiner. CosMiner is generally an FP-growth-like depth-first traversal algorithm that rests on an important property of the cosine similarity: the conditional anti-monotone property (CAMP). The combined use of CAMP and the depth-first support-ascending traversal strategy enables the pre-pruning of uninteresting patterns during the mining process of CosMiner. Extensive experiments demonstrate the high efficiency of CosMiner, in interesting pattern discovery, in comparison to the breath-first strategy and the post-evaluation strategy. In particular, CosMiner shows its capability in suppressing the generation of cross-support patterns and discovering rare but truly interesting patterns. Finally, an interesting case of landmark recognition is presented to illustrate the value of cosine interesting patterns found by CosMiner, in real-world applications.

1. Introduction

Since the introduction of frequent patterns in the early nineties of the last century [1], association analysis has become one of the core problems in the data mining and database communities [2]. Given a large set of items (objects) and the observation data about co-occurring items, association analysis is concerned with the identification of strongly correlated subsets of items [27]. It plays an important role in many application domains such as market-basket analysis [3,8], recommender systems [22], telecommunication network alarm analysis [26], image processing [25], climate studies [32], public health [10], and bio-informatics [38].

It was not until recently, researchers found that the classic support-confidence framework of association analysis has some intrinsic defects. For instance, the well-known “coffee–tea” example reveals that the confidence as an interestingness measure for association rules may not disclose truly interesting relationships [13,14]. Also, this framework often generates too many frequent patterns and even more association rules, from which to find useful patterns is itself a great challenge. Meanwhile, the support-based pruning strategy is not effective for data sets with skewed support distributions [40]; that is, if the minimum support threshold is relatively low, we may extract too many spurious patterns involving items with substantially different support levels, such as \{earrings,milk\} in which the support of earrings is much lower than that of milk. On
the contrary, if the threshold is high, we may miss many interesting but relatively infrequent patterns [16], such as \{earrings, gold ring, bracelet\} that contains rare but truly valuable items.

To overcome these problems, many interestingness measures have been proposed to fine truly interesting patterns [12]. Patterns with a measure value above some given threshold are called interesting patterns with respect to that measure. Among the proposed measures, the cosine similarity gains particular interests. Indeed, cosine similarity has various merits for association analysis. First, it is one of the few interestingness measures that hold symmetry, null-invariance and anti-cross-support-pattern properties simultaneously (details will be given in Section 2.2). Many well-known measures, such as confidence and lift, cannot achieve this. Second, cosine similarity has been widely used as proximity measure in text mining [41], information retrieval [17], image processing [11], and bio-informatics [20], to avoid the “curse of dimensionality” [4] – a problem we also face in association analysis. Finally, it is very simple and has physical meaning, i.e., it measures the angle of two vectors in the feature space. As a result, in this paper, we select the cosine similarity as the interestingness measure to define interesting patterns.

Mining cosine interesting patterns is by no means a trivial task. First of all, the traditional definition of cosine similarity is only for two vectors, which must be extended to the case of multi-itemsets. More importantly, the cosine similarity may not hold the anti-monotone property – the key for the success of frequent pattern mining using the support measure – and therefore may not be able to reduce the search space of interesting patterns. One may argue that a POST-evaluation scheme can already work, by first generating frequent patterns, and then identifying interesting patterns from frequent patterns using a cosine threshold. While being simple, the post-evaluation scheme has the same problem as the traditional frequent pattern mining algorithms; that is, rare but interesting patterns will be lost if a higher support threshold is set, but too many patterns will be generated for a low threshold instead.

In light of these, in this paper, we attempt to design an efficient algorithm to mine cosine interesting patterns from large-scale databases. To this end, we first expand the definition of cosine similarity to the scope of multi-itemsets, and highlight some valuable properties of the cosine similarity, such as the null-invariance property and the anti-cross-support-pattern property. We then point out that the cosine similarity does not hold the anti-monotone property, and propose a novel conditional anti-monotone property (CAMP) as an alternative. We argue that a depth-first itemset traversal strategy is more appropriate than a breath-first strategy when mining cosine interesting pattern using CAMP, based on which an algorithm named CosMiner, is proposed. CosMiner, is an FP-growth-like algorithm that uses the cosine as well as support thresholds to prune uninteresting patterns, although the mining of single-path tree is different from the FP-growth algorithm [15]. Extensive experiments on various real-world data sets demonstrate the superiority of CosMiner, to the Apriori-like method (CosMiner\_r) and the post-evaluation method (POST), in terms of efficiency. We also verify the capabilities of CosMiner, in suppressing cross-support patterns and mining the rare but interesting patterns. Finally, we apply CosMiner, as a noise-removal tool to a real-world landmark recognition case, based on which the performance of image clustering is improved substantially. This, in turn, demonstrates the value of cosine interesting patterns mined by CosMiner.

The remainder of this paper is organized as follows. In Section 2, we define and explore the cosine similarity from a pattern mining perspective. In Section 3, we propose the conditional anti-monotone property and suggest the combined use of a depth-first traversal strategy. The CosMiner algorithm is proposed in Section 4. Section 5 shows the experimental results, followed by a case study on landmark recognition in Section 6. We finally present the related work and conclude our work in Section 7 and Section 8, respectively.

2. Cosine similarity: preliminaries and problem definition

In this section, we present the definition of cosine similarity for multi-itemsets from an interestingness measure perspective. Some important properties of cosine similarity will be also discussed here. Finally, we formulate the problem to be studied in this paper.

2.1. The definition

Cosine similarity is typically regarded as a measure of proximity between two vectors. That is, given two vectors \( A \) and \( B \), it computes the cosine value of the angle between \( A \) and \( B \) as follows:

\[
\cos(A, B) = \frac{(A, B)}{|A||B|}.
\]

(1)

where “\( \langle \rangle \)” indicates the inner product of two vectors, and “\(||\)” indicates the length of a vector. For vectors with all non-negative features, the cosine value always varies in the range of \([0,1]\), where 1 indicates a perfect match of two vectors, whereas 0 indicates an absolute mismatch.

In the realm of associative analysis, if we replace \( A \) and \( B \) by items \( i_1 \) and \( i_2 \), respectively, in a transaction data set, the cosine similarity in Eq. (1) turns into

\[
\cos\{i_1, i_2\} = \frac{f_{i_1i_2}}{\sqrt{f_{i_1}f_{i_2}}},
\]

(2)
where \( f_{ij} \) is the number of transactions that contain both \( i \) and \( j \), and \( f_{i} \) and \( f_{j} \) are the numbers of transactions that contain \( i \) and \( j \), respectively. Let \( \text{supp}(X) \) denote the support value of a 2-itemset \( X = \{i, j\} \), the cosine similarity can be further regarded as an interestingness measure formulated by the support measure as follows:

\[
\cos(X) = \frac{\text{supp}(X)}{\sqrt{\text{supp}(\{i\}) \text{supp}(\{j\})}}.
\]

(3)

So far the cosine similarity is only for 2-itemsets. Nevertheless, we can naturally extend Eq. (3) to the multi-itemset case by averaging the support values of all the singleton itemsets in the denominator. We have the following definition:

**Definition 2.** For any itemset \( X = \{i_1, \ldots, i_K\}, K \geq 2 \in \mathbb{Z}_+ \), the cosine similarity of \( X \) is defined as

\[
\cos(X) = \frac{\text{supp}(X)}{\sqrt{\prod_{i=1}^{K} \text{supp}(\{i\})}}.
\]

(4)

Intuitively, cosine similarity measures the interestingness of an itemset by the co-occurrence intensity of all the items, conditioned on the occurrence of every single item. It therefore differs from the well-known support and confidence measures. The former does not take the condition into consideration, and the latter evaluates the interestingness of an association rule rather than an itemset. Note that cosine similarity was further generalized to the generalized mean measure in [37], where a conditional support concept was adopted for the measure derivation. Readers with interests along this line can refer to [37] for details.

### 2.2. The properties

Here, we present some properties of cosine similarity, which are crucial for understanding the strengths and weaknesses of employing cosine similarity for interesting pattern discovery.

Let \( T \) denote a transaction data set, and \( t \) denote a transaction in \( T \). The cosine similarity of an itemset \( X \) over \( T \) is denoted as \( \cos_T(X) \). Then we have the following property:

**Property 1 (Null-Invariance).** Suppose \( T \subseteq T' \) are two transaction data sets, and \( X \) is an itemset. If for all \( t \in T' \setminus T \), \( X \cap t = \emptyset \), we have

\[
\cos_T(X) = \cos_{T'}(X).
\]

(5)

The proof is straightforward using Eq. (4). The implication of the null-invariance property is that the value of \( \cos(X) \) is invariant if we add or delete transactions not containing any item of \( X \) to or from the data set. This is important, since the impact of data sizes can thus be isolated. In other words, by using cosine similarity, we can focus on the transactions that contain at least one item we have interests in. Note that we typically omit the subscript \( T \) in Eq. (5) in presenting the cosine similarity of an itemset when there is no confusion.

In real-world applications, many transaction data sets contain items with inherently skewed support distributions, which often lead to the so-called “cross-support patterns” (CSPs) defined as follows:

**Definition 2 (Cross-Support Pattern [39,40]).** Given a threshold \( \theta \in (0, 1) \), a pattern \( X \) is a cross-support pattern w.r.t. \( \theta \), if \( X \) contains two items \( x \) and \( y \) such that \( \frac{\text{supp}(x)}{\text{supp}(y)} < \theta \).

According to Definition 2, given \( \theta \in (0, 1) \), an itemset \( X = \{i_1, \ldots, i_k\} \) is a CSP if \( \frac{\min_{\text{supp}(\{i\})}}{\max_{\text{supp}(\{i\})}} < \theta \). This implies that a cross-support pattern typically contains items of substantially different support levels, i.e., frequencies of occurrences in transactions, and thus indicates spurious associative relationship. For instance, we may obtain a misleading itemset [cell phone, bread] from a supermarket transaction logs, just because the item bread is purchased by nearly everyone! Therefore, a good interestingness measure should be able to filter out CSPs, or equivalently, hold the Anti-Cross-Support-Pattern (Anti-CSP) property that suppresses the generation of CSPs. Fortunately, cosine similarity holds the Anti-CSP property, as formulated below:

**Property 2 (Anti-CSP).** For any itemset \( X = \{i_1, \ldots, i_k\}, X \) tends not to be a cross-support pattern as \( \cos(X) \) gets higher.

Property 2 is qualitative in nature. To illustrate this, let us first recall the computation of \( \cos(X) \) in Definition 1. Intuitively, the increase of \( \cos(X) \) implies the increase of \( \text{supp}(X) \) or the decrease of \( \sqrt{\prod_{i=1}^{k} \text{supp}(\{i\})} \). Since \( \text{supp}(X) \leq \min_{\text{supp}(\{i\})} \), the increase of \( \text{supp}(X) \) further implies the increase of \( \min_{\text{supp}(\{i\})} \). Moreover, since \( \sqrt{\prod_{i=1}^{k} \text{supp}(\{i\})} \) is the geometric mean of the item supports, its decrease implies the decrease of the item supports in general, and thus suggests the decrease of \( \max_{\text{supp}(\{i\})} \). To sum up, the increase of \( \cos(X) \) implies the increase of \( \min_{\text{supp}(\{i\})} \max_{\text{supp}(\{i\})} \), which by Definition 2 indicates that \( X \) tends not to be a cross-support-pattern. Note that the above reasoning is more qualitative than quantitative. In fact, there is no deterministic quantitative relationship between \( \cos(X) \) and \( \min_{\text{supp}(\{i\})} \max_{\text{supp}(\{i\})} \). As an important supplement, we provide empirical evidence in the experimental section below.
The anti-CSP property means that the cosine similarity can help to eliminate cross-support patterns, which is not pos-
sessed by many existing interestingness measures, such as the widely used support measure. We can also view this property
from another angle. In the proof, we actually point out a loose cosine upper bound for a cross-support pattern with respect to
$\theta : \sqrt{\theta}$. In other words, if we set $\min_{\cos} \geq \sqrt{\theta}$, $\forall X$ with $\cos(X) \geq \min_{\cos}$ will definitely not be a cross-support pattern. In
real-world applications, the CSP pruning effect of cosine similarity is indeed very significant. We present experimental
results in Section 5.

It is well recognized that to incorporate the interestingness measures into the pattern mining process, the measures are
desired to have the Anti-Monotone Property (AMP). Let $I$ be a universal itemset. A measure $M$ is anti-monotonic if
$\forall X, Y \subseteq I, X \subseteq Y \Rightarrow M(X) \geq M(Y)$. Very few measures holding AMP have been identified in the literature. For instance, the
support measure possesses the anti-monotone property, and therefore becomes the key for frequent pattern mining. The
other two measures are All-Conf (also called h-confidence in [40]) and Cohesion proposed in [24]. Regarding to the cosine
similarly, unfortunately, we have:

**Property 3 (Non-AMP).** The cosine similarity does not hold the anti-monotone property.

Recall Eq. (4). If two itemsets $X \subseteq Y$, we have $\text{supp}(X) \geq \text{supp}(Y)$ in the nominator. However, since the relative values of
the denominators w.r.t. $X$ and $Y$ are indeterminate, we cannot guarantee that $\cos(X) \geq \cos(Y)$. This can be illustrated by an
example. Assume a data set contains three transactions: $\{i_1, i_2, i_3\}$, $\{i_1\}$, and $\{i_2\}$. Let $X$ be $\{i_1, i_2\}$ and $Y$ be $\{i_1, i_2, i_3\}$, and thus
$X \subseteq Y$. According to Eq. (4), $\cos(X) = \frac{1}{\sqrt{2}+2} = 0.50 < \cos(Y) = \frac{1}{\sqrt{2}+2} = 0.63$, which indeed violates the anti-monotone
property.

In summary, after a natural extension, cosine similarity can serve as an interestingness measure for interesting pattern
(mining) evaluation, which possesses the valuable null-invariance and anti-CSP properties. However, the lack of the
anti-monotone property prevents it from being directly used as the support measure for interesting pattern mining. This,
indeed, motivates our study in this paper.

### 2.3 Problem definition

Given its appealing merits, we attempt to employ cosine similarity for interesting pattern discovery.

Let $\min_{\text{supp}}$ and $\min_{\cos}$ be the minimum thresholds of support and cosine similarity, respectively. An itemset $X$ is called
a cosine interesting pattern (CIP) with respect to a transaction data set $T$, if $\text{supp}_T(X) \geq \min_{\text{supp}}$ and $\cos_T(X) \geq \min_{\cos}$. Note
that we reserve the support measure in defining the cosine interesting patterns. This will help to divide CIPs roughly into two
categories, i.e., the significant CIPs with a relatively high $\min_{\text{supp}}$, and the rare CIPs with a relatively low $\min_{\text{supp}}$, both of
which may be of interests to users of different purposes.

The chokepoint here is, however, the cosine similarity does not hold the anti-monotone property. As a result, the Apriori
principle [1] established for the support measure cannot be leveraged to reduce the search space of CIPs. So our task in this
paper is summarized as following:

---

Find an alternative to the anti-monotone property for the efficient cosine interesting pattern discovery.

---

This task differs from the task of using cosine similarity for frequent pattern evaluation, although both tasks produce the
same results of cosine interesting patterns in theory. In fact, the evaluation scheme does not utilize $\min_{\cos}$ to prune uninter-

testing patterns during the pattern generation process, and therefore typically consumes much more time than the CIP
mining scheme. When mining rare but interesting patterns with a small $\min_{\text{supp}}$, the situation may be even worse, as the
evaluation scheme may collapse due to the unavailability of free memory for storing candidate frequent patterns. We
present the experimental results in Section 5.

### 3. Conditional anti-monotone property for interesting pattern mining

In this section, we present the Conditional Anti-Monotone Property (CAMP), an alternative to the anti-monotone prop-

erty, for cosine interesting pattern discovery. The itemset traversal schemes coupled with CAMP will be also discussed here.

#### 3.1 Definition of conditional anti-monotone property

We first define the novel conditional anti-monotone property as follows:

**Definition 3 (Conditional Anti-Monotone Property).** Let $I$ be a universal itemset. A measure $M$ holds the conditional anti-
monotone property, if $\forall X, Y \subseteq I$, given that (1) $X \subseteq Y$, and (2) if $Y \setminus X \neq \emptyset, \forall i \in X$ and $i' \in Y \setminus X$, $\text{supp}(i) \leq \text{supp}(i')$, we have
$M(X) \geq M(Y)$.
According to this definition, CAMP differs from AMP only in adding one more condition on the items in the difference set $Y \setminus X$. That is, the items in the difference set must have higher support values than the items in $X$. Based on this definition, we have:

**Theorem 1.** Cosine similarity holds the conditional anti-monotone property.

**Proof.** Without loss of generality, we assume that $X = \{i_1, \ldots, i_k\}$ is a $K$-itemset ($K \geq 1$), $Y = X \cup \{i_{k+1}, \ldots, i_{K+1}\}$ is a $(K+L)$-itemset ($L \geq 1$), with $\text{supp}(\{i_{k+1}\}) \geq \text{supp}(\{i_k\}), \forall 1 \leq l \leq L, 1 \leq k \leq K$. Now it remains to show $\cos(X) \geq \cos(Y)$.

Since $L \geq 1$, $Y \setminus X \neq \emptyset$. It is easy to note that $\text{supp}(X) \geq \text{supp}(Y)$. Moreover, according to the definition of geometric mean, we have

$$
\cos(X) = \frac{\text{supp}(X)}{\sqrt[K]{\prod_{k=1}^{K-1} \text{supp}(\{i_k\})}} \geq \frac{\text{supp}(Y)}{\sqrt[K]{\prod_{k=1}^{K-1} \text{supp}(\{i_k\})}} = \cos(Y),
$$

which completes the proof. $\square$

Note that CAMP can be regarded as a special case of AMP. Therefore, a measure possessing AMP certainly holds CAMP, such as the support measure. But the reverse is not true – cosine similarity is just a good example. Compared with the well-known AMP, CAMP demands an extra condition that the items in the difference set must have higher supports than the items in the subset. To achieve this goal, we must adopt a special itemset traversal sequence in the mining process. We detail this below.

### 3.2. Itemset traversal strategies

Given the universal itemset $\mathcal{I}$, the search space of interesting patterns consists of $2^{|\mathcal{I}|}$ different itemsets. This is often illustrated by a Hasse lattice [7]. Fig. 1(a) shows such an example with five items, where exactly 32 itemsets are generated in total.

To find the interesting itemsets in a lattice, a natural idea is to adopt a Breath-First Support-Ascending Traversal (BF-SAT) strategy. That is, the items in $\mathcal{I}$ are first sorted in a support-ascending order. We then traverse all the 1-itemsets, 2-itemsets, and so on, until all itemsets have been examined. Specifically, for any itemset $X$ including the empty set in the root, we only traverse its special immediate supersets generated by appending a single item in $\mathcal{I}$ to $X$, with the constraint that this item must have a rank lower than any of the items in $X$. Fig. 1(b) illustrates the BF-SAT scheme for five items from $A$ to $E$, where $\text{supp}(\{A\}) \leq \text{supp}(\{B\}) \leq \text{supp}(\{C\}) \leq \text{supp}(\{D\}) \leq \text{supp}(\{E\})$. Starting from the set of five items, BF-SAT will check all 2-itemsets, and then checks 3-itemsets, 4-itemsets and 5-itemsets successively, as shown in the marked region in Fig. 1(b). It is easy to note that BF-SAT is complete and correct – we can reach every itemset in Fig. 1(a) by traversing from the top to the bottom in a breath-first manner in Fig. 1(b), nothing more and nothing less.

Based on CAMP and BF-SAT, we have an Apriori-principle-like theorem as follows:

**Theorem 2.** For any interestingness measure holding CAMP, given the itemset traversal strategy BF-SAT, if an itemset is uninteresting, all its supersets directed by BF-SAT must also be uninteresting.

**Proof.** Assume $X$ is an uninteresting itemset with $\cos(X) < \min_\cos$. For any superset $Y$ of $X$, since the traversal strategy is BF-SAT, we have $\forall i \in X$ and $i' \in Y \setminus X, \text{supp}(\{i\}) \leq \text{supp}(\{i'\})$. Then according to CAMP, we have $\cos(Y) \leq \cos(X) < \min_\cos$. We complete the proof. $\square$

Given Theorem 2, it is natural to employ an Apriori-like algorithm for cosine interesting pattern mining. The only problem is that the traditional candidate generation strategy, i.e., $\mathcal{F}_k \times \mathcal{F}_k$, is no longer valid, with $\mathcal{F}_k$ being the set of interesting $k$-itemsets. This is due to the fact that cosine similarity possesses CAMP rather than AMP. For example, as indicated by Fig. 1(b), the fact that $(A, C)$ is not interesting indicates that its supersets $(A, C, D), (A, C, E),$ and $(A, C, D, E)$ are also uninteresting. But it does not indicate that $(A, B, C)$ is uninteresting, since $(A, B, C)$ is the immediate superset of $(A, B)$ rather than $(A, C)$ according to BF-SAT. Thus, we may miss the candidate $(A, B, C)$ due to the absence of $(A, C)$ in $\mathcal{F}_k$, if the $\mathcal{F}_k \times \mathcal{F}_k$ strategy is used. One possible remedy is to store the uninteresting but frequent $k$-itemsets for the generation of candidate interesting $(k + 1)$-itemsets during the mining process. This, although logically clear, certainly will increase the time and space complexity of the mining algorithm. We call this Apriori-like method CosMiner$_a$, and have implemented it for the experimental comparison in Section 5.

To overcome the problem of CosMiner$_a$, we here propose another itemset traversal strategy: DF-SAT (Depth-First Support-Ascending Traversal), and design an FP-growth-like algorithm called CosMiner, based on DF-SAT. In general, DF-SAT traverses all the itemsets in a depth-first manner, from the itemsets ended with the most infrequent item (with the smallest
support value), to the itemsets ended with the most frequent item (with the largest support value). Fig. 1(c) illustrates the DF-SAT scheme for five items from A to E, where \( \text{supp}(\{A\}) \leq \text{supp}(\{B\}) \leq \text{supp}(\{C\}) \leq \text{supp}(\{D\}) \leq \text{supp}(\{E\}) \). In this case, DF-SAT first traverses the itemsets ended with item A, and then the itemsets ended with items B, C, D and E successively.

For itemsets ended with item A, DF-SAT first examines the itemsets ended with \( \{B,A\} \), and then the itemsets ended with \( \{C,A\} \); \( \{D,A\} \) and \( \{E,A\} \), respectively. The traversal will be continued in this way, until all the itemsets have been examined.

Based on DF-SAT and CAMP, we also have an Apriori-principle-like theorem as follows:

**Theorem 3.** For any interestingness measure holding CAMP, given the itemset traversal strategy DF-SAT, if an itemset is uninteresting, all its supersets directed by DF-SAT must also be uninteresting.

The proof of Theorem 3 is very similar to that of Theorem 2, so we omit it here. Based on Theorem 3, it is not difficult to perceive that we can design an FP-growth-like method for cosine interesting pattern mining. We call this method CosMiner, which builds an FP-tree first by reading into memory the transactions with items sorted in a support-DESCENDING order, and then mines the cosine interesting patterns in a depth-first manner following the DF-SAT strategy. By Theorem 3, CosMiner can avoid examining the supersets of an uninteresting pattern, just like mining frequent patterns in an FP-tree using the support measure. Nevertheless, CosMiner also has some notable differences compared with the FP-growth algorithm. First, it requires the items in a transaction being sorted in a support-descending order before building the FP-tree, while FP-growth can accept any order, e.g., support-ascending order, support-descending order, or even the lexicon order, for constructing an FP-tree. Second, for a single-path tree, FP-growth can generate frequent patterns by simply doing combinations for the nodes in the tree. CosMiner, however, still needs to do sub-tree projections, since CAMP is weaker than AMP. We present the technical details in the next section.

**Discussions.** As the aforementioned, both CosMiner_a and CosMiner_t enable the cosine interesting pattern discovery. However, we argue that CosMiner_t is the default choice for the mining task due to the following reasons: (1) The candidate generation scheme of CosMiner_a is a bit complicated and may generate too many false-positive candidates, which is yet avoided by CosMiner_t; (2) CosMiner_t can be implemented easily upon the well-known FP-growth method, although several differences should be carefully addressed; and (3) the FP-growth method is generally faster than the Apriori method in
associative analysis [15], and this advantage can be inherited by CosMiner
. As a result, in what follows, we center on the
algorithmic details of CosMiner,
and provide comparative studies on CosMiner and CosMiner
in the experimental section.

4. CosMiner t: the algorithmic details

CosMiner t is generally an FP-growth-like algorithm for cosine interesting pattern discovery. It can be mainly divided into
two phases: (1) The generation of an FP-tree and (2) the mining of cosine interesting pattern from the FP-tree.

The first phase is almost the same as the FP-growth algorithm, except that the items in the transactions must be sorted in
a support-descending order before building the FP-tree. This is crucial for the implementation of the DF-SAT strategy, since
the patterns in an FP-tree grow forward by continuously appending one item as the prefix. Let us illustrate this by an exam-
ple shown in Fig. 3. As can be seen, there are eight transactions containing five items in the data set, and the support value
from E to A decreases successively. As a result, we must reorder the items in each transaction from E to A so that we can
obtain an FP-tree exactly as the one in Fig. 3. In this way, we guarantee that the conditional FP-tree of any itemset, such
as T_B of \{B\} in the figure, contains items all with support values greater than B. This implies that all the immediate supersets
of \{B\}, i.e., \{C, B\}, \{D, B\}, and \{E, B\}, will have cosine values no greater than \cos(B), and thus enables the DF-SAT strategy.

The second phase is for mining cosine interesting patterns (CIP) from the FP-tree. Fig. 2 sketches out the pseudocodes of
the mining process: CIP-growth. Especially to deserve to be mentioned, CIP-growth no longer distinguishes between single-
path trees and multi-path trees, which were however treated discriminatorily in the classic FP-growth procedure. This is be-
cause that for the single-path part, FP-growth simply returns all the combinations of items in as frequent patterns, due to the
powerful AMP. However, since the cosine similarity only possesses CAMP, CIP-growth cannot just enumerate the item com-
binations in the single-path tree as cosine interesting patterns. Instead, it needs to continue the sub-tree projection as for the
multi-path trees, although being much simpler – a single-path tree will be always projected to a single-path sub-tree. The
line 5 of Fig. 2 is for the sub-tree projections, for both single-path and multi-path trees.

For the sub-tree projection, FP-growth creates the conditional FP-tree for frequent pattern Y, but CIP-growth creates the
conditional FP-tree for interesting pattern Y only, which may reduce the database projections substantially. Comparative re-
sults along this line can be found in Section 5.3. Furthermore, CIP-growth has two notable technical details. First, a hash table
is constructed to store \(F_1\) list, in which a large support count is mapped to a small key value, for the purpose of accelerating
CIP-growth. Also, CIPs are stored in a hash table to keep the patterns in an orderly manner. Second, to avoid memory over-
flow, CIP-growth can promptly write the interesting patterns to the disk, although the running time will be increased
accordingly.

```
CIP-growth(Tree, X, min_supp, min_cos, \(F\))

Input:
Tree: the (conditional) FP-tree
X: the suffix pattern corresponding to Tree, and is \(\emptyset\) initially
min_supp: the minimum support threshold
min_cos: the minimum cosine threshold
\(F\): the set of CIPs, and is \(\emptyset\) initially

Procedure:
1. for each item \(i_k\) from bottom to top in Tree’s head table
2.    generate candidate CIP \(Y = \{i_k\} \cup X\);
3.    if \(\cos(Y) \geq \text{min_cos}\)
4.     \(F = F \cup Y\);
5.    create \(Y\)’s conditional FP-tree \(Tree_Y\) with min_supp;
6.    if Tree_Y \(\neq \emptyset\);
7.    CIP-growth(Tree_Y, Y, min_supp, min_cos, \(F\));
8.     \(\text{end}\)
9. \(\text{end}\)
10. \(\text{end}\)
11. return \(F\);
```

Fig. 2. Pseudocodes of CIP-growth.
4.1. Example

Fig. 3 shows a toy example to illustrate CIP-growth. The simulated data set contains eight transactions and five items. Let $\min_{\text{supp}} = 25\%$ and $\min_{\text{cos}} = 0.6$. Since the support count of $A, B, C, D, E$ are $3, 4, 5, 5, 7$, respectively, we have the support ascending order: $A \leq B \leq C \leq D \leq E$. The left-hand-side of Fig. 3 shows the data set, the head table and the generated FP-tree, respectively, and the right-hand-side shows the typical processes for mining the multi-path part (up-right) and the single prefix path (bottom-right), respectively.

At the beginning, the current item is the last item in the head table, i.e., $i_k = A$. Since the cosine similarity of one item is guaranteed to be 1, $A$’s conditional FP-tree $T_A$ is created. For $T_A$, the last item in its head table is $i_k = B$, and the current pattern suffix $X = \{A\}$. As we proceed, however, we will find that $Y = \{B, A\}$ is uninteresting for $\cos(Y) = 0.289 < 0.6$. So it does not need to construct $T_{BA}$ due to the conditional anti-monotone property. In contrast, if we turn to traversing $i_k = E$ in $T_A$, we will find that $Y = \{E, A\}$ is interesting with $\cos(Y) = 0.655$. But since the conditional FP-tree $T_{EA}$ is empty for $X = \{E, A\}$, we stop the deeper traversal.

Now assume we turn to traverse itemsets ended with $B$. Let $i_k = B$ and construct $B$’s conditional FP-tree $T_B$, which is a single prefix path shown in the bottom-right of Fig. 3. Then, we still scan $T_B$’s head table from bottom to up, i.e., $C, D, E$, and $F$, for sub-tree projections. For instance, in the first projection, since $\{C, B\}$ is uninteresting, we will stop the projection immediately. In the second projection, however, since $\{D, B\}$ is interesting, we will drill down until we find that $\{E, D, B\}$ is uninteresting. Mining itemsets ended with $\{E, B\}$ is similar to the situation when mining $\{E, A\}$ in $T_A$.

We finally obtain 7 interesting multi-item patterns: $\{E, A\}, \{D, B\}, \{E, B\}, \{D, C\}, \{E, D, C\}, \{E, C\}$, and $\{E, D\}$ after creating 7 conditional FP-trees only.

5. Experimental results

In this section, we provide experimental results on six real-world data sets. Two other mining algorithms, including the Apriori-like method (denoted as CosMiner$_a$) and the post-evaluation method (denoted as POST), are adopted for the comparative study. Note that CosMiner$_a$ employs a breath-first traversal strategy for interesting pattern mining, as described in Section 3.2. POST is more straightforward; that is, it first mines frequent patterns using the FP-growth algorithm, and then computes the cosine values of all frequent patterns to find the cosine interesting patterns, as described in Section 2.3. We will also look inside CosMiner, to explore the effect of the conditional anti-monotone property and the depth-first traversal strategy. Finally, we showcase the anti-cross-support-pattern property of CosMiner, and manifest some rare but truly interesting patterns found by CosMiner.

5.1. Experimental setup

5.1.1. Data

We used a number of real-world document data sets in our experiment. Some characteristics of these data sets are shown in Table 1, where “#Item” indicates the cardinality of the universal set of items, “AvgDim” indicates the average number of items contained by a record, and “Density” indicates the ratio of the non-empty elements in the data. These data sets were
obtained from various application domains. Data set Chess was from the game area and included in the UCI repository.\footnote{http://archive.ics.uci.edu/ml/} Accidents was donated by Karolien Geurts and contains anonymized traffic accident data. Pumb\_star corresponds to the binarized version of a census data set from IBM.\footnote{http://fimi.ua.ac.be/data/} Ohscal was obtained from the OHSUMED collection\footnote{http://fimi.ua.ac.be/data/}, which contains documents from various biological sub-fields. Product is a market-basket data set obtained from a large mail-order company. Kosarak was provided by Ferenc Bodon and contains anonymized click-stream data of a Hungarian online news portal.\footnote{http://www.borgelt.net/}

5.1.2. Tools
CosMiner\_t\footnote{http://www.borgelt.net/} and POST were coded upon the “FP-growth” source-codes provided by Borgelt,\footnote{http://www.borgelt.net/} and CosMiner\_a was coded upon “Apriori” source-codes obtained from the same source. All tools were coded in C, built by Visual Studio 2008, and run on a PC with an Intel 2.83 GHz CPU and 2 GB memory.

5.2. An overall comparison
Here we compare the efficiency of CosMiner\_t, CosMiner\_a and POST. Note that the I/O time for loading data and outputting patterns was not taken into account for the purpose of isolating the scale effect.

Fig. 4 shows the results of running time with \( \text{min}_\text{cos} \) values varying from 0.4 to 0.9. The \( \text{min}_\text{supp} \) threshold was carefully set for each data set to meet its very characteristic. As can be seen, CosMiner, generally shows much higher efficiency than the other two methods, and is far less sensitive to the increase of \( \text{min}_\text{cos} \) than CosMiner\_a. This implies that the combined use of the conditional anti-monotone property (CAMP) and the depth-first traversal strategy is superior for mining interesting

---

### Table 1
Experimental data sets.

<table>
<thead>
<tr>
<th>Data</th>
<th>Source</th>
<th>#Item</th>
<th>#Record</th>
<th>AvgDim</th>
<th>Density</th>
</tr>
</thead>
<tbody>
<tr>
<td>Chess</td>
<td>UCI</td>
<td>75</td>
<td>3196</td>
<td>37.0</td>
<td>0.4933</td>
</tr>
<tr>
<td>Accidents</td>
<td>NIS</td>
<td>572</td>
<td>340,183</td>
<td>33.8</td>
<td>0.0590</td>
</tr>
<tr>
<td>Pumb_star</td>
<td>IBM Almaden</td>
<td>2088</td>
<td>49,046</td>
<td>50.5</td>
<td>0.0242</td>
</tr>
<tr>
<td>Ohscal</td>
<td>OHSUMED-233,445</td>
<td>11,462</td>
<td>11,162</td>
<td>60.4</td>
<td>0.0053</td>
</tr>
<tr>
<td>Product</td>
<td>Mail-order company</td>
<td>14,462</td>
<td>57,671</td>
<td>7.2</td>
<td>0.0005</td>
</tr>
<tr>
<td>Kosarak</td>
<td>Ference Bodon</td>
<td>41,270</td>
<td>990,002</td>
<td>8.1</td>
<td>0.0002</td>
</tr>
</tbody>
</table>

---

Fig. 4. Runtime comparison along \( \text{min}_\text{cos} \).
patterns. Note that there is an exception in Fig. 4(e), where CosMiner, outperforms CosMiner, when min. cos is very large. This is due to the fact that the data set Kosarak has relatively few cosine interesting patterns (CIP) at high cosine levels, say only 125 CIPs with a maximum length of 3 when min. supp = 0.009% and min. cos = 0.9. In this situation, CosMiner, generates much less candidates, while CosMiner, suffers from the cost of constructing the FP-tree. This implies that CosMiner, might be particularly effective for mining large-scale and long interesting patterns. Finally, for POST, as the major time is spent on frequent pattern mining, it is almost insensitive to the change of min. cos, and in some cases, e.g., for data sets Pumsb_star and Product, works even faster than CosMiner, due to the adoption of the FP-tree.

We further investigate the impact of min. supp on the three algorithms. We loosely set min. cos = 0.6 to avoid pruning too many patterns. Fig. 5 shows the running time results along the increasing min. supp value. Again we can observe that CosMiner, is still the most efficient one on almost all the data sets, though the gaps narrow quickly with the increase of min. supp. It is also noteworthy that, for some data sets, e.g., Chess, Accidents, Product and Kosarak, POST or CosMiner, may crash due to the memory overflow when a small min. supp value is set, as indicated by the missing points in Figs. 5(a), (b), (d) and (e), respectively. This implies that to mine interesting but RARE patterns, which is often with a very small support value, CosMiner, may be the best choice among the three methods.

It is noteworthy that different support thresholds were set for various data sets in Fig. 4 and support thresholds in Fig. 5 do not vary linearly. This is because the support thresholds do depend on the unique properties of the data sets. A larger value may result in too few patterns, but a small value may lead to too many spurious patterns, both of which are undesirable. Therefore, we conducted a lot of experiments using multiple support thresholds, and chose an suitable support threshold for each data set in Fig. 4 and some points in Fig. 5, in order to better illustrate the major trends of the impact. The same treatment can be found in Ref. [9,29].

To sum up, CosMiner, is generally faster than CosMiner, and POST in mining cosine interesting patterns. Moreover, it is particularly suitable for mining large-scale or rare interesting patterns.

5.3. Inside CosMiner,: the effect of CAMP

Here, we attempt to quantify the effect of the conditional anti-monotone property (CAMP) on CosMiner,. As an FP-growth-like algorithm, the core operation of CosMiner, is to recursively compute the conditional FP-tree for a projected data set. Therefore, the reduced number of projections from POST to CosMiner, can well describe the major effect of CAMP.

Fig. 6 compares the number of projections between CosMiner, and POST on each of the six data sets, where min. cos is set to 0.6 by default. As can be seen, for all the data sets, CosMiner, has far less projections than POST. This indicates that although with a not-very-large min. cos value, CAMP already shows significant effect on reducing the projections in CosMiner,. As min. supp increases, however, the support measure rather than the cosine similarity gradually becomes the dominant factor, and the gaps narrow accordingly.

![Fig. 5. Runtime comparison along min. supp.](image-url)
5.4. Inside CosMiner: the effect of the anti-CSP property

In this section, we show the effect of CosMiner on reducing the cross-support patterns (CSP). We first introduce the ratio of deleted cross-support patterns ($r$) as follows:

$$r = \frac{\text{#deleted cross-support patterns}}{\text{#cross-support patterns}},$$

where the denominator indicates the number of total cross-support patterns given a specific threshold $\theta$, and the nominator denotes the accumulated number of deleted CSPs given specific support and cosine thresholds.

We investigate the impact of $\text{min}_\text{cos}$ on randomly selected three data sets: Pumsb_star, Product, and Accidents. For each data set, we set three different values to $\theta$ for testing the robustness of the results, and then observe the variation of $r$ with the increase of $\text{min}_\text{cos}$. As can be seen from Fig. 7, it is a general trend that $r$ increases rapidly as $\text{min}_\text{cos}$ increases gradually. When $\text{min}_\text{cos}$ reached 0.7, over 90% of the cross-support patterns are filtered out for the three data sets, with the only exception for Product with $\theta = 0.5$. This observation indicates that the anti-CSP property of cosine similarity is significant in real-world applications, which better ensures the true interestingness of the discovered patterns.

5.5. Inside CosMiner: finding rare but interesting patterns

In this section, we examine the interestingness of patterns found by CosMiner. We explain this from two perspectives as follows.

Fig. 6. CosMiner, vs. POST: the number of projections.

Fig. 7. Effect of $\text{min}_\text{cos}$ on reducing cross-support patterns.
First, CosMiner enables the search for rare but interesting patterns. By using \( \min_{\cos} \) as the threshold, we can set a lower \( \min_{\text{supp}} \) value such that more rare but interesting patterns can be found without increasing the total number of patterns dramatically. To illustrate this, we set \( \min_{\text{supp}} = 0 \) and \( \min_{\cos} = 0.6 \) for CosMiner, to mine all frequent or rare interesting patterns from data sets Product and Ohscal. Fig. 8 exhibits the accumulative distribution of the support value of the found interesting patterns. As can be seen, for both data sets, the majority of interesting patterns have quite small support values, which may probably get lost in a traditional frequent pattern mining process, where a relatively high support threshold is usually set to avoid the overload of patterns.

Then we judge the real interestingness of the found patterns by the semantics of the items in the patterns. Table 2 shows some sampling interesting patterns with very low support values identified by CosMiner, from both the Product and Ohscal data sets. As can be seen, the items in these patterns are strongly correlated, and tend to appear simultaneously in the transactions (records). For instance, in the Product data set, the first pattern containing two kinds of shams, one drape and two kinds of pillows is about household goods, and the second pattern \{breadbox, soup, rack, covers, mug\} is about kitchen goods. For the Ohscal data set containing texts from the biological area, the pattern \{hexamethylen, diisocyan, isocyan, toluen\} is mainly about the organic compounds, and the pattern \{kearn sayre, mtDNA\} contains various kinds of DNA.

In summary, the traditional frequent pattern mining algorithms may not discover interesting patterns with low support values, since reducing the support threshold to a low level may lead to the exponential growth of patterns. However, by incorporating the cosine similarity as the interestingness measure, CosMiner, can find these rare but truly interesting patterns effectively.

6. An application

In this section, we employ CosMiner, as a massive-noise removal tool, to enhance the cluster analysis for high-dimensional landmark recognition.

6.1. Data

The landmarks are contained in the Oxford_5K data set\(^5\), which is an image collection retrieved from Flickr.\(^6\) In total, it contains 5060 images of 11 different Oxford landmarks – a landmark here means a particular part of a building. Each image is then scanned for salient regions, and a high-dimensional descriptor is computed for each region. These descriptors are then quantized into a vocabulary of visual words. As a result, an image is represented by a bag of 1 M visual words, with feature values being the numbers of word occurrences. Table 3 shows some characteristics of this image collection, where the features occurring less than three times were removed. As can be seen, Oxford_5K is a very sparse high-dimensional data set. Moreover, one of four possible labels was manually assigned to each image \([30]\), i.e., “Good”, “OK”, “Bad”, and “Junk”, where images labeled “Good” or “OK” indicated 25% of the object is clearly visible. We then merged landmarks labeled “Good” and “OK”, and treated them as clear images. We finally obtained only 568 clear images. In other words, roughly 88% images in Oxford_5K are noise in nature!

\(^5\) http://www.robots.ox.ac.uk/~vgg/data/oxbuildings/index.html.
6.2. Procedure

Given the Oxford_5K data set, our task is to do image clustering, and then identify the eleven Oxford landmarks from the clusters. To this end, we must deal with two challenges from the high sparseness of data and the vast noise inside data. The former can be addressed by adopting a clustering tool designed purposefully for high-dimensional data, such as the well-known CLUTO [23]. To address the latter, we use the proposed CosMiner, as a noise-removal method here. That is, we first transform Oxford_5K into a transaction data set, and then employ CosMiner on the new data to mine cosine interesting patterns (CIPs). In this case, a CIP is a set of visual words that occur simultaneously in some images. Images that do not cover any CIP will be treated as noise, and therefore removed from the data. Fig. 9 shows the complete procedure for the landmark recognition, where in the "Image clustering" phase the cluster number was set to 11 for vcluster (an algorithm in CLUTO) with default settings, and in the "Landmark recognition" phase we manually removed the remaining noise and labeled a cluster by the dominant landmark in that cluster.

6.3. Result

Table 4 shows the parameter settings and the clustering performance. In the table, "#CIPs" indicates the number of interesting patterns discovered by CosMiner, "#Images" indicates the number of remaining images after noise removal, "#Clear_images" indicates the number of non-noisy images among the remaining images, and "NMI" is a widely used external measure for clustering evaluation [33]. NMI is in the range of [0, 1], and a larger NMI indicates a better clustering performance.

One observation is that, without noise removal, the clustering quality in Scene 1 is extremely poor due to the very bad influence exerted by the vast noise. Another observation is that min_cos is a key factor that determines the effect of noise removal. A small min_cos will lead to more CIPs, and thus less noise removals, which cannot improve the clustering quality substantially, as indicated by Scene 2. However, a large min_cos may lead to excessive noise removal, and thus increase the risk of removing too many normal instances as well, which will also degrade the clustering performance, as indicated by Scenes 5 and 6. In contrast, by setting min_cos to 0.48 in Scene 4, the clustering quality of CLUTO is improved significantly.

Table 2
Sampling interesting patterns found by CosMiner.

<table>
<thead>
<tr>
<th>Data set</th>
<th>Interesting patterns</th>
<th>Support (%)</th>
<th>Cosine</th>
</tr>
</thead>
<tbody>
<tr>
<td>Product</td>
<td>sham/FLNCE, sham/SCLP, drape, pillow/NCKRL_pillow/SQ</td>
<td>0.005</td>
<td>0.600</td>
</tr>
<tr>
<td>Product</td>
<td>breadbox, soup, rack, covers, mug</td>
<td>0.007</td>
<td>0.695</td>
</tr>
<tr>
<td>Product</td>
<td>valance, curtain, charm, grease</td>
<td>0.007</td>
<td>0.730</td>
</tr>
<tr>
<td>Product</td>
<td>nokia battery, nokia adapter, nokia wireless phone</td>
<td>0.048</td>
<td>0.690</td>
</tr>
<tr>
<td>Product</td>
<td>doll/M_FEB, doll/M APR, doll/M MAY, doll/JUNE, doll/JULY, doll/AUG, doll/SEPT, doll/M OCT, doll/M DEC</td>
<td>0.024</td>
<td>0.913</td>
</tr>
<tr>
<td>Ohscal</td>
<td>hexamethylen, disocyan, isocyan, toluen</td>
<td>0.018</td>
<td>0.620</td>
</tr>
<tr>
<td>Ohscal</td>
<td>neuroradiograph, myelo, HNP (herniated nucleus pulposus), pulposu</td>
<td>0.018</td>
<td>0.620</td>
</tr>
<tr>
<td>Ohscal</td>
<td>aminopropan, doi iodophenyl, spiperon</td>
<td>0.027</td>
<td>0.641</td>
</tr>
<tr>
<td>Ohscal</td>
<td>ther, pharmacol, exp</td>
<td>0.027</td>
<td>0.754</td>
</tr>
<tr>
<td>Ohscal</td>
<td>learn sayre, mtDNA</td>
<td>0.027</td>
<td>0.693</td>
</tr>
</tbody>
</table>

Table 3
Oxford_5K data set.

<table>
<thead>
<tr>
<th>Dataset</th>
<th>#Instances</th>
<th>#Features</th>
<th>#Class</th>
<th>Density</th>
<th>Noise</th>
</tr>
</thead>
<tbody>
<tr>
<td>Oxford_5K</td>
<td>5060</td>
<td>658,346</td>
<td>11</td>
<td>0.0228%</td>
<td>88.22%</td>
</tr>
</tbody>
</table>

Fig. 9. Overview of procedure for landmark recognition.
We then identify the landmarks from the clusters generated in Scene 4. Altogether 9 out of 11 landmarks were successfully identified. Fig. 10 depicts 10 images found in the cluster for each of the four sample landmarks: All_souls, Christ_church, Ashmolean, and Corn_market. Among them, All_souls and Christ_church are two relatively easy cases. For instance, in the cluster of Christ_church, 50 out of 51 images contain the correct landmark, which comes near to perfection. In contrast, the other two landmarks, i.e., Ashmolean and Radcliffe_camera, are much harder to identify. For instance, the cluster of Ashmolean contains 13 instances, but only 8 images are the correct ones. Finally, it is noteworthy that two landmarks, i.e., Corn_market and Keble, were not recognized from the clusters due to the rareness of the samples – only 1 and 3 images contain these two landmarks, respectively, in the data set.

To sum up, using CosMiner as a booster for noise removal, we successfully identify nine out of eleven landmarks from extremely sparse and noisy image data. This case clearly illustrates the practical value of cosine interesting patterns discovered by CosMiner.

7. Related work

Since the early introduction by Agrawal et al. [1], association analysis has been widely used in many application domains [13,35]. But one problem is that the traditional support–confidence framework tends to generate too many rules – many of them are indeed uninteresting to us.

To cope with this problem, many interestingness measures have been proposed or borrowed to mine the truly interesting patterns. Piatetski–Shapiro proposed the statistical independence of rules as an interestingness measure [31]. Brin et al. [6] proposed lift and $\chi^2$ as correlation measures and developed an efficient mining method. Omiecinski [28] and Lee et al. [24] found that all-confidence, coherence, and cosine were null-invariant and thus good measures for mining correlation rules in transaction databases. Blanchard et al. [5] designed a rule interestingness measure, Directed Information Ratio, based on information theory. Hilderman and Hamilton [19] and Tan et al. [34] provided well-organized comparative studies for the

<table>
<thead>
<tr>
<th>Scene</th>
<th>Parameter setting</th>
<th>#CIPs</th>
<th>#Images</th>
<th>#Clear_images</th>
<th>NMI</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>min_supp</td>
<td>min_cos</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>–</td>
<td>–</td>
<td>–</td>
<td>5060</td>
<td>568</td>
</tr>
<tr>
<td>2</td>
<td>0.09</td>
<td>0.45</td>
<td>14,816</td>
<td>1788</td>
<td>546</td>
</tr>
<tr>
<td>3</td>
<td>0.09</td>
<td>0.47</td>
<td>13,495</td>
<td>1009</td>
<td>370</td>
</tr>
<tr>
<td>4</td>
<td>0.09</td>
<td>0.48</td>
<td>12,675</td>
<td>1001</td>
<td>368</td>
</tr>
<tr>
<td>5</td>
<td>0.09</td>
<td>0.50</td>
<td>11,428</td>
<td>994</td>
<td>227</td>
</tr>
<tr>
<td>6</td>
<td>0.09</td>
<td>0.60</td>
<td>7770</td>
<td>771</td>
<td>183</td>
</tr>
</tbody>
</table>

Fig. 10. Four sample landmarks recognized using CosMiner, and CLUTO.
interestness measures, respectively, and Han et al. [13] provided a comprehensive review for the interestingness measures in the whole framework of frequent pattern mining. Recently, Wu et al. [37] used the notion of the generalized mean to generalize some interestingness measures including the cosine similarity. However, they did not study how to use the cosine similarity in the mining algorithms.

A key problem in using the interestingness measures above is the lack of the anti-monotone property, which makes it difficult to incorporate the measures directly into the mining algorithms. Nevertheless, people still identified some measures holding the anti-monotone property as follows. Lee et al. [24] proposed the CoMine algorithm for the mining of all-confidence and coherence patterns. Xiong et al. [40,21] proposed the hyperclique mining algorithm for the efficient discovery of hyperclique patterns. They also pointed out that the h-confidence measure was equivalent to the all-confidence measure. Wu et al. [36] firstly incorporated the cosine measure as an IN-evaluation measure, and thus presented an Apriori-like algorithm CosMiner using a breath-first method. Besides cosine similarity, we have found some other interestingness measures such as all-confidence, Kulc, h, and MaxConf also hold the CAMP. Naturally, all of these interestingness measures can be seamlessly incorporated into our CosMiner, as the IN-evaluation measure.

In summary, despite of the vast amount of research efforts in interesting pattern mining problem, further study is still needed to find an efficient way to incorporate the cosine similarity into the interesting pattern mining process. Our work in this paper just aims to fill this crucial void.

8. Conclusions

This paper studied the problem of mining cosine interesting patterns from large-scale databases. The conditional anti-monotone property as well as the depth-first traversal strategy were proposed to elicit a novel FP-growth-like algorithm: CosMiner. Extensive experiments with comparative studies demonstrated the strengths of CosMiner. As an application example, CosMiner was successfully applied to a landmark recognition task where huge volume of image noise was presented.

Acknowledgments

This research was partially supported by the National Natural Science Foundation of China (NSFC) under Grants 71072172, 71372188 and 61103229, National Center for International Joint Research on E-Business Information Processing under Grant 2013B01035, National Key Technologies R&D Program of China under Grant 2013BAH16F00, the National Soft Science Research Program under Grant 2013GXXS4B801, Industry Projects in Jiangsu S&T Pillar Program under Grant BE2012185, and the Natural Science Foundation of Jiangsu Province of China under Grant BK2012863. Prof. Junjie Wu’s research was supported in part by NSFC under Grants 71322104, 71171007 and 71031001, by the Foundation for the Author of National Excellent Doctoral Dissertation of PR China under Grant 201189, and by the Program for New Century Excellent Talents in University under Grant NCET-11-0778.

References

[13] Jiawei Han, Hong Cheng, Dong Xin, Xifeng Yan, Frequent pattern mining: current status and future directions, Data Min. Knowl. Discov. 15 (1) (2007) 55–86.
[14] Jiawei Han, Micheline Kamber, Jian Pei, Data Mining: Concepts and Techniques, second ed., Morgan Kaufmann, 2005.
[15] Jiawei Han, Jian Pei, Yiwen Yin, Mining frequent patterns without candidate generation, in: Proceedings of the 2000 ACM SIGMOD International Conference on Management of Data, Dallas, TX, 2000, pp. 1–12.
[37] Tianyi Wu, Yuguo Chen, Jiawei Han, Association mining in large databases: a re-examination of its measures, in: Proceedings of the 11th European Conference on Principles and Practice of Knowledge Discovery in Databases, 2007, pp. 621–628.