Under consideration for publication in Knowledge and Information Systems

Diverse Dimension Decomposition for Itemset Spaces

Mikalai Tsytsarau[1], Francesco Bonchi[2], Aristides Gionis[2] and Themis Palpanas[1]
[1]University of Trento, Italy; [2]Yahoo! Research Barcelona, Spain.

Abstract. We introduce the problem of diverse dimension decomposition in transactional databases, where a dimension is a set of mutually exclusive itemsets. The problem we consider requires to find a decomposition of the itemset space into dimensions, which are orthogonal to each other and which provide high coverage of the input database. The mining framework we propose can be interpreted as a dimensionality-reducing transformation from the space of all items to the space of orthogonal dimensions.

Relying on information-theoretic concepts, we formulate the diverse dimension decomposition problem with a single objective function that simultaneously captures constraints on coverage, exclusivity and orthogonality. We show that our problem is **NP**-hard and we propose a greedy algorithm exploiting the well-known FP-tree data structure. Our algorithm is equipped with strategies for pruning the search space deriving directly from the objective function. We also prove a property that allows assessing the level of informativeness for newly-added dimensions, thus allowing to define criteria for terminating the decomposition.

We demonstrate the effectiveness of our solution by experimental evaluation on synthetic datasets with known dimension and three real-world datasets, `flickr`, `del.icio.us` and `dblp`. The problem we study is largely motivated by applications in the domain of collaborative tagging, however, the mining task we introduce in this paper is useful in other application domains as well.

Keywords: Diversity, transactional databases, dimension decomposition, information theory, collaborative tagging

1. Introduction

Collaborative content creation and annotation is one of the main activities and distinguishing features of the Web 2.0. Advances in social-media and user-

Received Dec 24, 2011
Revised May 03, 2012
Accepted May 13, 2012

t_1	{fish,art,film,portrait,tattoo,xpro,crossprocessed,nikon,skin,n80}
t_2	{sanfrancisco,black&white,building,art,stairway,fireescape,nikon}
t_3	{portrait,color,art,me,illustration,blood,adobephotoshop,canon}
t_4	{travel,brazil,plant,art,nature,color,strong,nikon,nikond70}
t_5	{sunset,art,museum,landscape,minneapolis,canon,powershotg3}
t_6	{sculpture,art,2004,festival,japan,culture,clay,a70,canon}
t_7	{portrait,art,painting,color,europe,sony,sonyericssonk750}
t_8	{black&white,art,film,photograph,street-photo,contax645}
t_9	{art,black&white,skin,hand,bodypainting,nikon,d70}
t_{10}	{red,woman,art,face,color,tear,canon,eos300d}
t_{11}	{art,3d,unfound,photositook,sony,cybershot}
t_{12}	{beautiful,woman,black&white,portrait,art}
t_{13}	{landscape,nature,sunrise,wallpaper,art}

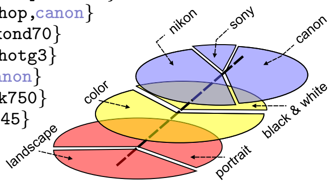

Fig. 1. An example of transactional dataset, having three diverse dimensions (shown on the right). In this specific example from Flickr, each transaction corresponds to a picture, and its associated tags. All pictures have in common the tag `art`.

generated content technologies have resulted in collecting extremely large volumes of user-annotated media; for instance photos (`flickr`), urls (`del.icio.us`), blogs (`technorati`), videos (`youtube`), songs (`last.fm`), scientific publications (`bibsonomy` and `citeulike`), and others. All these platforms provide users with the capability of generating content and assigning *tags*, i.e., freely chosen keywords, to this content.

A repository of tagged resources can be seen as a transactional database, typical to the paradigm of frequent-itemset mining: transactions correspond to resources, and items correspond to tags. In this setting we are interested in studying the problem of discovering an item-space decomposition, which we define to be a set of orthogonal dimensions with high coverage. A dimension in turn is defined to be a set of itemsets that rarely co-occur in the database.

Example 1. Consider for instance a query on `flickr` for photos about art (i.e., annotated with the tag `art`): the dataset $\mathcal{D}$ of such photos can look like the one in Figure 1. In this setting dimensions might be, for example, such sets of itemsets as {{`portrait`},{`landscape`}} or {{`canon`},{`nikon`},{`sony`}}. Indeed, almost all photos in the dataset contain at most one of the itemsets from each of the two dimensions.[1]

In particular, we are interested in discovering dimensions that represent diverse concepts, such as "type of photo" or "camera brand", and whose different values almost partition the dataset. For instance, each dimension in Figure 1 can be seen as a different way of partitioning the transactions in $\mathcal{D}$, and the three dimensions together can be considered as a diverse decomposition of the space of photos.

Extracting a set of diverse and orthogonal dimensions, that covers the various aspects of the underlying dataset, can be seen as a problem of *automatic facet discovery*, with plenty of applications in social-media and user-generated content platforms. Whenever users label resources with tags, the automatic discovery of the relevant dimensions can improve user experience, e.g., facet-drive browsing and exploration, faceted search [11], tag recommendation [10], tag clustering [12, 13, 14, 15], and more. Similarly in recommender systems, whatever

[1] While in this example each dimension is formed by singleton items, in general a dimension is formed by itemsets of any size.

is the subject of the recommendation (movies, songs, books, etc.) structuring the recommendation by means of automatically discovered facets (i.e., orthogonal dimensions) can improve the users experience and facilitate the discovery of interesting resources. In the scientific bibliography domains, discovering diverse and orthogonal dimensions can help highlighting the main trends in a scientific community over a period of time.

Towards our goal, we adopt an information-theoretic perspective. While there exist several studies applying *joint entropy* to the problem of identifying interesting or informative itemsets [1, 2, 4, 5, 6, 7], this body of work can not be applied to the problem of diverse dimension decomposition, as explained next. Joint entropy is conventionally calculated as a sum of entropies over probabilities (frequencies) of set's instances: $H(X) = -\sum p(X_i) \log p(X_i)$, where instances $X_i \subseteq X$ are often represented as binary vectors, where bits at each position are indicating whether a particular item is present in the instance or not. In the following example we demonstrate, that items in such sets are usually semantically unrelated:

Example 2. Consider a transposed view of the database from Figure 1, as shown in Table 1. Following the approaches that use *joint entropy*, we will get sets (templates) such as {`color,nikon`}, having the highest entropy (dark grey lines), or {`landscape,sony`} as low-entropy sets (light grey lines).

We notice that high-entropy sets are characterized by more uniform appearance of their instantiations in the database (e.g., instances 01, 10 and 11 appear with roughly the same frequency), while low-entropy sets accumulate support around the few most-frequent instances (in our example: 00), not necessarily representing mutually exclusive items forming the dimension (with instances 001, 010, 100). Thus, using the existing interestingness measures does not solve our problem.

In this article, we propose an entropy measure that expresses both the orthogonality among dimensions and the interestingness of dimensions. Moreover, we show that the proposed measure also captures constraints on exclusivity and coverage. Based on this measure, we formulate diverse dimension decomposition as the problem of finding an optimal set of k dimensions, minimizing an objective function that closely resembles the mutual information measure, except for a parameter α, which allows the analyst to trade-off between information loss and orthogonality of the dimensions.

Our contributions are summarized as follows.

- We introduce the novel problem of diverse dimension decomposition in transactional databases, as an optimization problem. We characterize our objective function and show that the selected dimensions explain well the underlying database.
- We prove a property that allows assessing the level of informativeness for newly-added dimensions, thus allowing to define criteria for terminating the decomposition.
- We show that our problem is **NP**-hard. We then propose a greedy algorithm exploiting the well-known FP-tree data structure [8]. Our algorithm prunes the search space, based on properties of our objective function.
- We experiment with the proposed approach using two real-world large datasets in the collaborative tagging domain, `flickr` and `del.icio.us`, and one dataset

Table 1. A transposed view of the dataset in Figure 1, showing most frequent items taken from several dimensions.

item	t_1	t_2	t_3	t_4	t_5	t_6	t_7	t_8	t_9	t_{10}	t_{11}	t_{12}	t_{13}
`canon`	0	0	1	0	1	1	0	0	0	1	0	0	0
`nikon`	1	1	0	1	0	0	0	0	1	0	0	0	0
`sony`	0	0	0	0	0	0	1	0	0	0	1	0	0
`color`	0	0	1	1	0	0	1	0	0	1	0	0	0
`black&white`	0	1	0	0	0	0	0	1	1	0	0	1	0
`landscape`	0	0	0	0	1	0	0	0	0	0	0	0	1
`portrait`	1	0	1	0	0	0	1	0	0	0	0	1	0

in the scientific bibliography domain (`dblp`) demonstrating the effectiveness and scalability of our solution.

– We introduce two methods, which generate synthetic datasets with distributions of itemsets closely approximating the real data. Experiments on these datasets indicate that our method is able to withstand noise and reliably identify dimensions even with small coverages.

– We evaluate, both theoretically and empirically, the ability of various itemset similarity measures to facilitate the dimension decomposition. Moreover, we outline several crucial properties for creating such measures.

The present article extends an earlier conference version [9] and has an updated formalism. In particular, itemset frequency is used instead of itemset support count, and Theorem 2 is reformulated using ordered conditional entropies. The rest of the article is structured as follows. In the next section we discuss related work and in Section 3 we formally define the problem of mining diverse dimensions from a transactional dataset. In Section 4 we present our methods, while in Section 5 we report experimental assessment. Finally, we discuss future work and conclude in Section 6.

2. Related work

We next survey the literature related to our work, dividing it into three groups: (*i*) methods that aim at extracting diverse content from web data, (*ii*) space-like representations of itemset databases, and (*iii*) entropy-based measures for itemset interestingness.

2.1. Diversity in information retrieval

Extracting a set of diverse dimensions, that covers the various aspects of the underlying dataset, can be seen as a problem of facet discovery. Such a facet-discovery process has many applications in improving user experience, for instance, tag recommendation [10], search and exploration [11], tag clustering [12, 13, 14, 15], and more. The work by Morik et al. [15], describing multi-objective hierarchical termset clustering, represents an effective way of navigating tag-annotated resources. However, the hierarchical clusters produced in the above work do not represent orthogonal dimensions, which is one of our main goals.

Web search is another domain in which finding an answer set with diversity is important. Several studies have focused on the problem of search engines query-result diversification [16, 17, 18, 19], where the goal is to produce an answer set that includes results relevant to different aspects (facets) of the query.

2.2. Space-like representation of itemset databases

Traditionally, in association-rule mining, itemsets are represented as binary vectors in the space of items: each axis corresponds to an item, and binary coordinate values indicate whether each particular item is contained in the itemset. This representation works well, if we are interested in finding association rules of the form {`bread,milk`} ⇒ {`butter`}, which capture itemset-level correlations in data. However, binary coordinates do not facilitate geometric decompositions of the item space (which can be interpreted by a human).

As a possible solution, Korn et al. [20] use real-valued coordinates, where coordinates can be interpreted as quantities of each item employed in the construction of rules. This framework allows to perform spectral decomposition of the item space (similar to SVD [21]), and to discover *ratio rules*, i.e., quantitative correlations among itemsets. An example of such rule is {`1:bread,2:milk, 5:butter`}, which says that a typical ratio of bread, milk and butter within the itemsets is {`1:2:5`}, so we can predict missing values of different items given these rules.

Alternatively, one can represent a database in the transposed space of transactions rather than items (like the one shown in Table 1). This is the main idea behind the "geometrically inspired itemset mining" framework proposed by Verhein and Chawla [22]. Their proposal is a framework for frequent itemset mining, which can accept space transformations, such as SVD, subject to the constraint that a measurement function should be able to be computed in the new space. For instance, in the case of SVD, each new axis represents a linear combination of transactions, featuring the largest variance in data. However, such a transformation is not very easily interpretable.

Our work is different in that we propose a method for decomposing the space of items in a set of orthogonal dimensions that are readily interpretable. Moreover, our problem formulation is based on information theory, and is capable of identifying dimensions in transactional databases in general, regardless whether transactions have real values associated with items or not.

2.3. Entropy-based measures of itemset interestingness

Knobbe and Ho [1] define an information-theoretic measure for itemset interestingness, *joint entropy*, which is optimizing for uniform co-occurrence among items. In their terminology, a set is a template (or a collection of attributes taking binary values), whose instances are itemsets. Entropy is calculated as a negative sum of logarithm-multiplied occurrence probabilities for observed instances. This measure indicates how likely a randomly-chosen set instance is to appear in data. The same authors also introduced a notion of "pattern teams" [2], that can be seen as feature sets. They theoretically evaluate the effectiveness of different filtering criteria for feature sets used in machine-learning classifiers, noticing that the measure of joint entropy does not satisfy the desirable properties that we require from dimensions in this paper (i.e., mutual exclusivity, high coverage).

Table 2. Summary of notation used in this paper.

I	$I \in \mathcal{I}$ is an item	X	$X \subseteq \mathcal{I}$ is a pattern itemset
$\mathcal{I}$	$\mathcal{I} = \{I\}$ is a set of all items	δ^i	$\delta^i = \{X^i\}$ is i-th dimension
t	$t \subseteq \mathcal{I}$ is a transaction itemset	Δ	$\Delta = \{\delta^i\}$ is a set of dimensions
$\mathcal{D}$	$\mathcal{D} = \{t\}$ is a dataset of transactions	$\mathbb{I}$	is a space of all possible itemsets: $t \in \mathbb{I}$

Instead, the authors find that *exclusive coverage* (i.e., the sum of coverages minus co-occurrences) is much more suitable as a measure optimizing for these properties.

Continuing the above line of research, Heikinheimo et al. define two related problems, namely, mining high- and low-entropy sets [6]. Zhang and Masseglia [7] extend the method of Heikinheimo et al. to work on streaming data. They propose to reduce the output size by removing similar sets according to criteria based on mutual information [23]. A similar approach for counting subset frequencies was also used by Tatti [3] to define the significance (surprisingness) of itemsets, by comparing their Maximum-Entropy estimated frequency to the observed one.

Finally, Tatti [4] and Mampaey et al. [5] propose to use joint entropy in an MDL optimization framework, aiming at compressing the database. Maximizing the entropy ensures that all the pattern subsets are uniformly distributed, while the limit on pattern frequency (according to the exponential frequency decrease assumption) facilitates the selection of frequent patterns.

Although these papers deal with itemset mining using joint entropy, their goal is different from ours: they aim at extracting sets of items, which co-occur in the database uniformly (when optimized for high entropy: same frequency for all subset combinations) or sparsely (when optimized for low entropy: only certain subsets are frequent). We discussed the difference between these approaches and our proposal earlier, in Example 2.

In contrast to the above methods, we formulate the entropy of a dimension as the uncertainty of the dimension's itemsets for each document, and use it as an indicator of quality for dimensions. Moreover, our goal is to find sets of itemsets (not items), which are not only mutually exclusive (within each dimension), but also independent (across dimensions).

3. Notation, concepts, and problem statement

We are given a transactional dataset $\mathcal{D}$, which is a multi-set of transactions $t \subseteq \mathcal{I}$, where $\mathcal{I}$ is a ground set of items. An example of a transactional dataset is given in Figure 1. As usual we call itemset any set of items $X \subseteq \mathcal{I}$, and we denote by $\mathcal{D}(X)$ its supporting set of transactions, i.e., $\mathcal{D}(X) = \{t \in \mathcal{D} \mid X \subseteq t\}$. Moreover we denote by $\mathbb{I}$ the space of all possible itemsets on $\mathcal{I}$. For the convenience, we are presenting a short list of notation in Table 2.

In this paper we are studying the following problem. We are given an integer k and the goal is to discover a collection of k dimensions, that decompose the itemset space $\mathbb{I}$. Moreover, we want each dimension to almost partition the dataset $\mathcal{D}$; that is to say, we want (almost) all transactions $t \in \mathcal{D}$ to contain one and only one of itemsets from the dimension.

Definition 1 (Dimension). Given an itemset space $\mathbb{I}$, a dimension $\delta^i \subset \mathbb{I}$ is a collection of pairwise disjoint itemsets, i.e., $\delta^i = \{X_0^i, ..., X_m^i\}$, such that for all pairs of itemsets $X_k^i, X_l^i \in \delta^i$ with $l \neq k$ it holds $X_k^i \cap X_l^i = \emptyset$.

As in decomposition methods in linear algebra, we want to decompose the itemset space in dimensions that can be though as "orthogonal." While orthogonality in linear algebra is a well-understood concept, when talking about the itemset space the concept of orthogonality is much less clear. Motivated by our example, we would like to argue that the dimension representing *camera-brand* {{`canon`},{`nikon`},{`sony`}, ...} is orthogonal to the dimension *type-of-photograph* {{`portrait`},{`landscape`},{`street-photo`}, ...}. The concept of orthogonality can thus be formulated as independence among the dimensions: the fact that a photograph is tagged by `nikon` should not reveal any information about the type of the photograph. That is, the likelihood of that photograph being `portrait` or `landscape` should remain the same as it is non conditional on *camera-brand.*

To formalize the above intuition, we use the concept of *mutual information.* Given two random variables, X and Y, mutual information measures the information shared between them. For example, if X and Y are independent, then knowing X does not give any information about Y and vice versa, so their mutual information is zero. In order to employ the definition of mutual information, we need to define precisely how our dimensions define a probability space, and what is the entropy of this probability space. We provide those definitions in the next section.

In addition to finding orthogonal dimensions we also want to find "useful" dimensions, in the sense of being able to explain the dataset succinctly. We express this intuition by the concept of *coverage.* In the previous example, the dimension *camera-brand* has high coverage because most of the photos have one tag among the itemsets {{`canon`},{`nikon`}, ...}. We are able to show that the concept of coverage can also be formulated in an information theoretic manner. More importantly, we are able to combine both desiderata, high coverage and orthogonality, in one single objective function, achieving to simplify our problem formulation as well as the mining algorithm.

3.1. Entropy of dimensions

Our goal is to define the entropy $H(\delta^i \mid \mathcal{D})$ of the dimension $\delta^i = \{X_0^i, ..., X_m^i\}$ of the itemset space $\mathbb{I}$, given the input dataset $\mathcal{D}$. We first define the entropy of the dimension δ^i conditioned on a single transaction t of the dataset.

$$H(\delta^i \mid t) = -\sum_{X^i \in \delta^i} P(X^i \mid t) \log P(X^i \mid t)$$

The probabilities $P(X^i \mid t)$ express the uncertainty that the itemset X^i is present in the transaction t, and are defined later in the section. Averaging over all transactions of the dataset $\mathcal{D}$, we now define the entropy of the dimension δ^i as follows:

$$H(\delta^i \mid \mathcal{D}) = \sum_{t \in \mathcal{D}} H(\delta^i \mid t) P(t),$$

where $P(t)$ is the frequency of each transaction in the dataset. For instance, if

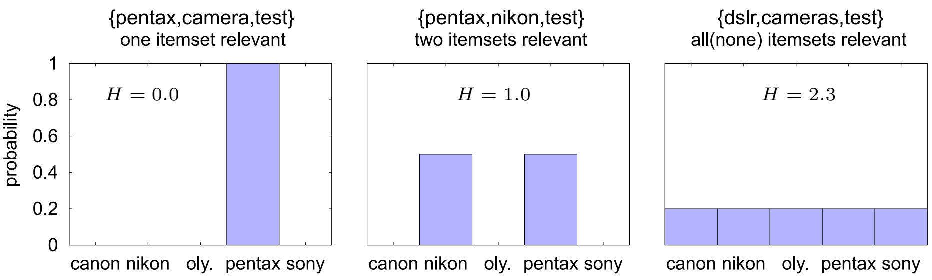

Fig. 2. Entropy for different probability distributions.

all transactions are distinct, then $P(t) = 1/|\mathcal{D}|$. The conditional entropy of one dimension given another, is defined as follows:

$$H(\delta^i \mid \ \delta^j, \mathcal{D}) = \frac{1}{|\delta^j|} \sum_{X^j \in \delta^j} \frac{H(\delta^i \mid X^j, \mathcal{D})}{\sum_{t \in \mathcal{D}} P(X^j \mid t)}, \quad \text{where}$$

$$H(\delta^i \mid X^j, \mathcal{D}) = \sum_{t \in \mathcal{D}} H(\delta^i \mid X^j, t) P(X^j \mid t) P(t), \quad \text{and}$$

$$H(\delta^i \mid X^j, \ t) = -\sum_{X^i \in \delta^i} P(X^i \mid X^j, t) \log P(X^i \mid X^j, t).$$

It remains to define the probabilities $P(X^i \mid t)$, which can be interpreted as the probability of an itemset being relevant for a transaction. When computing relevancy probabilities, we may use different set similarity measures, such as *cosine similarity*, *Jaccard coefficient*, or *binary inclusion/exclusion*:

$$P(X^i \mid t) = \frac{|X^i \cap t|}{||X^i|| \, ||t||}, \quad P(X^i \mid t) = \frac{|X^i \cap t|}{|X^i \cup t|}, \quad \text{or} \quad P(X^i \mid t) = \begin{cases} 1, & X^i \subseteq t; \\ 0, & X^i \nsubseteq t. \end{cases}$$

Also note that after computing the set similarity measures we need to normalize them in order to arrive to a valid probability distribution whose values sum up to 1. The following example describes the meaning of different probability distributions.

Example 3. Let us consider a dimension δ^i containing five itemsets: {{`canon`}, {`nikon`},{`olympus`},{`pentax`},{`sony`}}. Each transaction t in the dataset may be relevant to one or several itemsets of the dimension, or not relevant at all. Figure 2 shows three different transactions with the following probability distributions: t_1 = {`pentax, camera, test`} is relevant only to {`pentax`}, with probability 1.0; t_2 = {`pentax, nikon, test`} is relevant to two cameras, with probability 0.5; t_3 = {`dslr, cameras, test`} may be relevant to any camera, thus resulting in equal probabilities and maximal entropy. In this example, entropy reflects the uncertainty of the dimension being relevant to a transaction. When only one itemset is relevant we have low entropy, as in the first case. When none of the itemsets is relevant, resulting in the unclear choice, the entropy becomes high.

3.2. Problem statement

As we mentioned before, the problem we consider is to discover k diverse dimensions that explain well the input dataset. Let us denote by $\Delta = \{\delta^1, \dots, \delta^k\}$ such a set of k dimensions. Our objective function evaluates the goodness of the dimension set Δ in terms of entropy and diversity. We define these concepts next.

Definition 2 (Entropy of a dimension set). Given a set of dimensions $\Delta = \{\delta^1, \dots, \delta^k\}$, its entropy is defined as the sum of entropies of its dimensions[2]

$$H(\Delta) = \sum_{\delta^i \in \Delta} H(\delta^i).$$

Definition 3 (Diversity of a dimension set). Given a set of dimensions $\Delta = \{\delta^1, \dots, \delta^k\}$, its diversity is defined as the sum of conditional entropies over all pairs of dimensions

$$DIV(\Delta) = \sum_{\delta^i, \delta^j \in \Delta} H(\delta^i \mid \delta^j).$$

Central to our problem is the concept of mutual information, which we define here for a pair of dimensions δ^i and δ^j.

Definition 4 (Mutual Information). The mutual information of two dimensions δ^i and δ^j is defined as follows

$$I(\delta^i; \delta^j) = H(\delta^i) - H(\delta^i \mid \delta^j) = H(\delta^j) - H(\delta^j \mid \delta^i).$$

The mutual information of two dimensions is symmetric and is computed by taking the difference between an entropy of the first dimension, $H(\delta^i)$, and its conditional entropy given another one, $H(\delta^i \mid \delta^j)$. The latter entropy expresses the amount of information which one dimension contains about another, and we want this amount to be low (this happens when the conditional entropy of dimension δ^i remains large after we have identified dimension δ^j). In order to evaluate the goodness of the set of dimensions Δ we are summing the mutual information among all pairs of dimensions of the set Δ. We are now ready to formally define our problem.

Problem 1 (Diverse Dimension Decomposition). Given a dataset $\mathcal{D}$, find a set of k dimensions Δ that minimize $f(\Delta)$:

$$f(\Delta) = \left[H(\Delta) - \frac{\alpha}{k-1} DIV(\Delta) \right]. \quad (1)$$

In the above problem definition, we propose using an optimization function $f(\Delta)$ derived from mutual information.

Additionally, we introduce a parameter α to control the effect of entropy and conditional entropy over the optimization criterion. One can notice that the value of $\alpha = 1$ corresponds to the case when the criterion is based precisely on

[2] Throughout our paper we assume that all entropies are calculated with respect to the dataset $\mathcal{D}$, omitting it in order to simplify the notation.

the pairwise sum of mutual informations, but we may pick any other positive real value. This gives us the possibility to optimize either for information loss (when α is small, e.g., $\alpha = 0$), orthogonality (when α is large, e.g., $\alpha = 1$), or for both (when α takes an intermediate value).

Furthermore, we are able to show that by minimizing the objective function (1) we are also ensuring that the resulting dimensions explain well the underlying dataset. We first define the notion of coverage of a dimension.

Definition 5 (Coverage of a dimension). Coverage $C(\delta)$ of the dimension δ on the dataset $\mathcal{D}$ is the fraction of transactions t in $\mathcal{D}$, for which $t \cap X \neq \emptyset$, for some itemset $X \in \delta$.

Definition 6 (Maximal co-occurrence of a dimension). We define the maximal co-occurrence $R(\delta)$ of the dimension δ on the dataset $\mathcal{D}$ as the fraction of transactions t in $\mathcal{D}$ which contain all the itemsets from a subset $\{X\} \in \delta$, and none of the other subsets of δ are more frequent.

The following two lemmas are needed in our exposition that minimizing $f(\Delta)$ ensures high coverage.

Lemma 1. If the value of the objective function is less than a threshold, $f(\Delta) \leq \psi$, then the entropy of the dimensions is also bounded:

$$H(\Delta) \leq \frac{\psi}{1-\alpha}.$$

Proof. For all pairs of dimensions δ^i and δ^j, we have that $H(\delta^i \mid \delta^j) \leq H(\delta^i)$, what implies that $I(\delta^i; \delta^j) \geq 0$. In the case of a pairwise sum, $DIV(\Delta) \leq (k-1)H(\Delta)$. Consequently, if $[H(\Delta) - \alpha DIV(\Delta)/(k-1)] \leq \psi$ we have that $[H(\Delta) - \alpha H(\Delta)] \leq \psi$, or equivalently, $H(\Delta)(1-\alpha) \leq \psi$. □

The lemma implies that for values of $\alpha \geq 1$ the entropy becomes unbounded. In other words, when optimizing solely for orthogonality the quality (entropy) of dimensions may become uncontrollable as some of them can be added to a collection solely because of their high independence to others. This can happen for dimensions that have negative contributions to $f(\Delta)$ because of a high value of the parameter α.

Property 1. For a dimension δ, let s be the number of co-occurring itemsets in a transaction t, where $2 \leq s \leq |\delta|$. Then, in the case of binary probabilities

$$P(X \mid t) = \frac{1}{s}, \text{ and } H(\delta \mid t) = -s\frac{1}{s}\log\frac{1}{s} = \log s,$$

where in the case of no-coverage we have $s = |\delta|$, since all itemsets are equally improbable.

Lemma 2. Let δ be a dimension with m itemsets, and consider the case that the probabilities $P(X \mid t)$ take binary values. Then for the coverage $C(\delta)$ of the dimension δ it should be

$$C(\delta) \geq 1 - \frac{H(\delta)}{\log m}.$$

Proof. Entropy takes its maximum value in the case that a transaction is not covered by a dimension δ. Applying Property 1, we have that $\max(H(\delta \mid t)) = \log m$. Therefore, the maximal ratio of not covered transactions would be less than $H(\delta)$ divided by the maximum entropy. Thus, $1 - C(\delta) \leq H(\delta)/\log m$, what proves the lemma. □

Lemma 3. If probabilities $P(X \mid t)$ are computed using binary similarities, then maximal co-occurrence R between any two itemsets in a dimension δ should be less than its entropy per single transaction: $R(\delta) \leq H(\delta)$.

Proof. Property 1 shows that the minimal entropy of single co-occurrence is equal to $\log 2$. The maximal relative number of how many times the two itemsets may co-occur would be $H(\delta)$ divided by the minimal entropy of single co-occurrence. Therefore, we have $\max(R(\delta)) = H(\delta)/\log 2 = H(\delta)$. □

We are now stating the theorem that small values of $f(\Delta)$ imply high coverage. The theorem is a direct consequence of Lemmas 1 and 2.

Theorem 1. Let $\Delta = \{\delta^1, \ldots, \delta^k\}$ be a set of k dimensions and $C(\Delta)$ be their total coverage, defined as $C(\Delta) = \sum_{\delta^i \in \Delta} C(\delta^i)$. Finally, let m_0 be the size of the smallest dimension of Δ. If $f(\Delta) \leq \psi$ then for the total coverage we have:

$$C(\Delta) \geq k - \frac{\psi}{(1-\alpha)\log m_0}.$$

Proof. According to Lemma 2, the sum of dimensions coverages is greater than:

$$\sum_{\delta^i \in \Delta} C(\delta^i) \geq k - \sum_{\delta^i \in \Delta} \frac{H(\delta^i)}{\log m} \geq k - \sum_{\delta^i \in \Delta} \frac{H(\delta^i)}{\log m_0}$$

Applying our notation and using Lemma 1, we have:

$$C(\Delta) \geq k - \frac{H(\Delta)}{\log m_0} \geq k - \frac{\psi}{(1-\alpha)\log m_0} \qquad \square$$

We can use the above theorem to evaluate the quality of the dimensions, or to limit the number of dimensions in the result, e.g., by conforming to the specified constraint on the minimum coverage.

We next evaluate the dependency of $f(\Delta)$ over the number of dimensions k. Suppose that we have a set of dimensions, and want to add another dimension.

Theorem 2. Adding a candidate dimension δ will improve $f(\Delta)$ as long as its average mutual information (across dimensions Δ) is less than the average information of dimensions in Δ discounted by $(\frac{1}{\alpha} - 1 + \frac{1}{k})H(\delta)$.

Proof. The difference d in the optimality value can then be written as follows:

$$\begin{aligned} d &= H(\delta) - \frac{\alpha}{k} DIV(\Delta \cup \delta) + \frac{\alpha}{k-1} DIV(\Delta) \\ &\leq H(\delta) - \frac{\alpha}{k}[DIV(\Delta \cup \delta) - DIV(\Delta)] \\ &\leq H(\delta) - \frac{\alpha}{k} \sum_{\delta^i \in \Delta} [H(\delta|\delta^i) + H(\delta^i|\delta)] \end{aligned}$$

We are interested in cases when this difference will be negative, corresponding to improving optimality:

$$H(\delta) - \frac{\alpha}{k} \sum_{\delta^i \in \Delta} [H(\delta|\delta^i) + H(\delta^i|\delta)] \leq 0, \text{ and since}$$

$$H(\delta|\delta^i) + H(\delta^i|\delta) = H(\delta) + H(\delta^i) - 2 \cdot I(\delta;\delta^i), \text{ we have}$$

$$H(\delta) - \alpha\frac{k-1}{k}H(\delta) - \frac{\alpha}{k}\sum_{\delta^i \in \Delta} H(\delta^i) + \frac{2\alpha}{k}\sum_{\delta^i \in \Delta} I(\delta;\delta^i) \leq 0, \text{ equivalent to}$$

$$\frac{1}{k}\sum_{\delta^i \in \Delta} I(\delta;\delta^i) \quad \leq \quad \frac{1}{2}[\frac{1}{k}H(\Delta) - (\frac{1}{\alpha} - 1 + \frac{1}{k})H(\delta)] \qquad \square$$

In other words, $f(\Delta)$ will decrease when dimensions in Δ on average contain less information about δ than their average information discounted by $H(\delta)$. As we noted before, since $I(\delta;\delta^i)$ is always greater than zero, smaller α would require addition of dimensions with smaller $H(\delta)$ to satisfy this condition, thus optimizing for information loss. This property allows assessing the level of informativeness for newly-added dimensions, and defining criteria for terminating the decomposition.

4. Algorithm

We observe that Diverse Dimension Decomposition (Problem 1) is **NP**-hard, by reduction from the Set Partitioning problem, where we want to partition a set into non-overlapping and non-empty parts that cover the entire set. This inherent complexity of the problem motivates us to seek for a heuristic algorithm. In the rest of this section, we describe our solution based on a greedy strategy.

4.1. Algorithm overview

In order to solve the optimization problem of finding the optimal k dimensions, we propose identifying dimensions one-by-one. Our greedy strategy works as follows. We start by constructing the first more prominent dimension, according to our objective function $f(\Delta)$. The process begins with an empty single dimension, and at each iteration we decide whether to add a new dimension, or grow existing itemsets, according to the strategies discussed below. The construction of each dimension stops either if it is not possible to improve its score or if all items have been partitioned. Then, we do the same for the remaining dimensions iteratively, with the only difference that $f(\Delta)$ now takes into account all the previously identified dimensions, optimizing with respect to their orthogonality.

To store the data for our problem, we adopt a compressed database representation in the form of the well-known FP-Tree data structure [8]. In Figure 3, we show an example of such a tree, for the transactional dataset of Figure 1. This structure allows us to perform efficient pruning based on the coverage, co-occurrence and non-overlap (partitioning) requirements, as explained next.

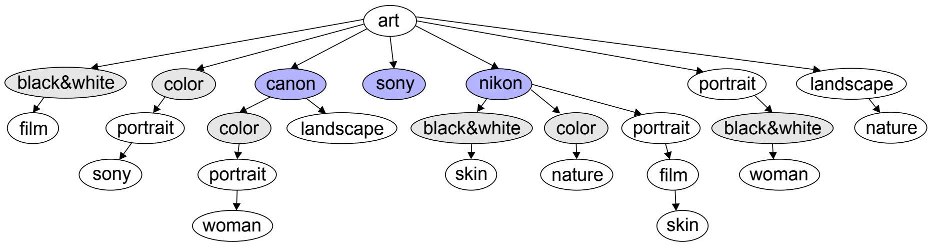

Fig. 3. FP-Tree constructed from the dataset shown in Figure 1. For the sake of simplicity, we omitted frequency counts from the nodes, cross-references among nodes and the header table, showing only the prefix-tree. To avoid having a too large figure, the tree is shown after a pruning of itemsets of frequency less than 2. Nodes highlighted in light gray represent items from first-order dimensions, which are blocked and become transparent when considering itemsets for a new dimension (highlighted in blue).

4.2. Search strategies

We now discuss the search strategies that can be used over the FP-tree data structure, as well as the pruning techniques that can be applied on top of those.

– **Breadth-first strategy (*expansion*)**: (a) Locate, and remove from further consideration, individual nodes for items that are already in the dimension (according to the non-overlap criterion; for example, nodes, highlighted in gray in Figure 3); (b) add one of the remaining available singleton items as a new itemset; we add these items one at a time.

– **Depth-first strategy (*refinement*)**: (a) For an itemset in the dimension, locate the correspondent paths in the FP-tree; (b) Expand this itemset by adding one item at a time from the available children nodes of its paths.

However, the problem with the above strategies is that neither of them can lead to a good solution, when used independently: the breadth-first strategy may include many singleton items so that refinement (or expansion) of individual itemsets in a dimension is no longer possible; the depth-first strategy may restrict adding new itemsets to the dimension by expanding existing itemsets with their children items.

– **Mixed strategy (*expansion + refinement*)**: Apply the expansion and refinement steps at every iteration. This is the strategy we use in this paper, and we discuss it in more detail in the following paragraphs (refer to Algorithm 1).

4.3. Pruning strategies

We have already described the basic pruning strategy (non-overlap) based on our definition for dimensions. Our more advanced pruning strategy is based on the relationship between entropy and such characteristics as *coverage* and *co-occurrence*, as described in Lemmas 2-3. For each candidate dimension with entropy H, we are interested in obtaining refined dimensions, which do not exceed this value. Thus, we compute the corresponding thresholds for the minimal coverage C and maximal co-occurrence R (according to the above lemmata), and use them for pruning the itemsets which are added or refined.

Algorithm 1: Mining Orthogonal Dimensions

```
   Name  : findNewDimension
   Input : First-order dimensions $\Delta = \{\delta^k\}, k < i$,
           Candidate dimensions $candidates = \{\}$,
           FP-Tree, memoryBudget
   Output: Optimal dimension $\delta^i_{out}$ of order $i$
1  repeat
2      forall the dimension $\delta^i \in candidates.unprocessed$ do
3          forall the itemset $X^i \in \delta^i$ do
4              forall the items $I \in children(X^i),\ I \notin \delta^i, \Delta$ do
5                  if validItemset($X^i \cup \{I\} \mid \delta^i$) then
6   refinement         //add one item to the current itemset
7                      $\delta^i_{temp} = \{\delta^i \mid X^i = X^i \cup \{I\}\}$;
                       checkOptimality($\delta^i_{temp} \mid \Delta$);
                       $candidates.temp.add(\delta^i_{temp})$;
8                  end
9              end
10         end
11         forall the items $I \in \mathcal{I},\ I \notin \delta^i, \Delta$ do
12             if validItemset($\{I\} \mid \delta^i$) then
13  expansion      //add one more item as an itemset
14                 $\delta^i_{temp} = \delta^i \cup \{I\}$;
15                 checkOptimality($\delta^i_{temp} \mid \Delta$); $candidates.temp.add(\delta^i_{temp})$;
16             end
17         end
18     end
19     //mark unprocessed as processed
20     candidates += candidates.unprocessed;
21     //newly generated become unprocessed
22     candidates.unprocessed = candidates.temp;
23     candidates.temp = {};
24     //sort so that most optimal values are first
25     candidates.sort();
26     //remove candidates exceeding the allocated memory
27     repeat
28         candidates.remove(candidates.lastElement);
29     until $candidates.size > memoryBudget$;
30 until $candidates.unprocessed.size > 0$;
31 return $\delta^i_{out} = candidates.firstElement$;
```

4.4. Description of the algorithm

We formulate our optimization problem in a greedy fashion, relying on a mixed candidate generation strategy and an iterative refinement of the candidate set. The complexity of this approach (almost) linearly depends on the size of the candidate set (as seen in Figure 5), which we use as a parameter. Another input of our algorithm is the FP-Tree, optionally containing only the most frequent

Algorithm 2: ItemSet pruning method `validItemset`. Entropy for a given dimension will improve only if the new itemset meets coverage and co-occurrence requirements computed using Lemmas 2-3 for the dimension's entropy.

```
Name   : validItemset
Input  : Dimension $\delta^i$, itemset $X$, FP-Tree
Output: true if itemset is valid, false otherwise
1 //use Lemma 2 to calculate min coverage
2 $\text{cov}_{\text{min}} = \texttt{Lemma2}(H(\delta^i))$;
3 //use Lemma 3 to calculate max co-occurrence
4 $\text{cooc}_{\text{max}} = \texttt{Lemma3}(H(\delta^i))$;
5 return ($\texttt{C(}X \cup \delta^i\texttt{)} > \text{cov}_{min}$ & $\texttt{R(}X, \delta^i\texttt{)} \leq \text{cooc}_{max}$);
```

items. This initial pruning does not affect the output (as long as the items forming the dimensions are preserved), but significantly reduces the complexity of the problem.

Our general approach starts with an empty set of dimensions, and uses Algorithm 1 to find each new dimension, resulting in the best optimality value when added to the set of previously selected dimensions; up to the specified number k.

The most essential part of this algorithm is the greedy dimension optimization procedure `findNewDimension`, which takes as a parameter a set of the first-order dimensions Δ, and an empty set of candidates, and after a finite number of iterations (the first loop) it converges to the single most optimal dimension, which is added to the Δ as the next one.

More specifically, Algorithm 1 iteratively refines dimensions in the candidate set (the empty initial set is refined only by expansion) and at each iteration performs sorting of candidates according to their optimality. The list of sorted dimensions is then being pruned according to the specified memory budget. By doing this operation, the algorithm ensures that at each step it would refine and check the optimality of only a short list of candidates, which is equal to the memory budget or lower. After all candidates in the list were refined, they are marked as processed (transferred to the main list), and the newly generated list of candidates becomes the next list of unprocessed candidates. The algorithm converges when there are no candidates left in the list, which were not refined. Then it outputs the topmost optimal candidate.

More insights on the algorithm can be obtained by examining Figure 4. In that figure, we depict detailed results of the refinement procedure for two specific domains, namely, "pyramid" and "art", from the `flickr` dataset. In both cases, we focus on the identification of the first dimension, and we depict for each iteration of the Algorithm 1 the size (number of itemsets) of the currently best dimension (bottom graphs), as well as the corresponding entropy value (top graphs).

We observe that for the "pyramid" domain the algorithm quickly increases the size of the dimension by adding more itemsets (seen as diagonal steps) and refining them (seen as horizontal steps), resulting in a significant initial improvement of the entropy of the dimension. Starting at iteration 6, the number of itemsets remains stable, though, the algorithm adds new items to them, leading to further improvements in entropy (which can be observed by the decrease of value on the top-most graph). In contrast, for the "art" domain the algorithm

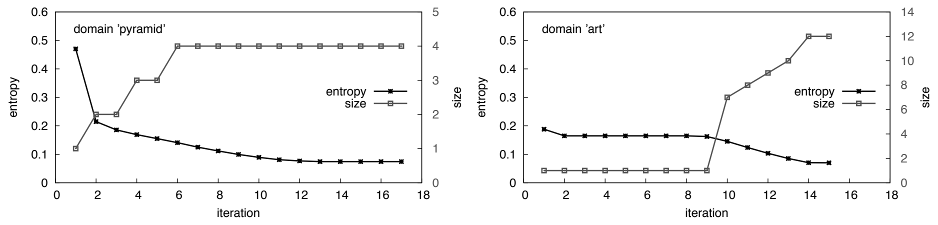

Fig. 4. Optimization statistics of the first dimension for "pyramid" and "art" (`flickr`).

starts with a dimension of good quality (low entropy), which after a single refinement (from iteration 1 to 2) stays on the top of the list of candidates till iteration 9 (the value of entropy does not increase during this interval), while other candidates are being refined. Then, starting from the iteration 10, another candidate, refined to a better quality, takes its place and shows even better improvement in entropy. Finally, iterations converge and the best dimension is being identified.

We note that the final dimensions identified for the "pyramid" and "art" domains (after 17 and 15 iterations, respectively) are also the optimal single dimension decompositions for these domains.

5. Experimental evaluation

We evaluate our algorithm on both synthetic[3] and real-world datasets. Evaluation of precision and performance is done in a controlled scenario, using large-scale synthetic datasets with artificially generated dimensions having itemset distribution similar to that of real datasets. We evaluate the time of decomposition and the quality of extracted dimensions depending on various noise and coverage parameters and for several versions of our method. The observed results convince that our method achieves good performance and is capable of reconstructing dimensions even in most unfavorable conditions.

We provide additional evidence regarding the usability of our method by applying it to two real-life datasets, containing tag-annotated resources, and one dataset, containing titles of scientific publications. The first dataset, extracted from `flickr`, a popular photos sharing website, contains 28 million tag-sets (or transactions), obtained by taking annotations for all pictures that contain a specific domain tag, for 34 different domains. To remove noise, we allow only unique tag-sets for each user id. The second dataset contains tag-sets from `del.icio.us`, a social bookmarking website. For this dataset, we select annotations for URLs starting with specific domain names picked from `Yahoo! Directory`. Overall, the `del.icio.us` dataset contains 1.7 million tag-sets over 150 domains. The number of unique tags in each of the datasets is about half a million. The third dataset, `dblp`[4], is the largest collection of publications in the area of Computer Science, containing more than 1.9 million titles organized by venue and year. In our experiments, we used publications titles as sets of tags, removing stop-words, punctuation and duplicate words.

[3] Synthetic datasets are available at http://disi.unitn.it/~tsytsarau/#content=Datasets
[4] DBLP database dump is publicly available at http://dblp.l3s.de/dblp++.php

A limited amount of additional cleaning is performed on all datasets by removing the domain term, numeric and navigational tags, as well as removing some language variability, based on a custom-built dictionary. No sophisticated preprocessing is applied, so some of the discovered dimensions in our experimental results still contain repetitions due to synonyms and misuse of tags.

We implement our algorithms in Java, and run the experiments under Java JRE 1.6.13 on a machine with Intel Xeon 2.4 GHz processor, using a single core and 1Gb of allocated memory.

5.1. Quantitative results on real datasets

In the first set of experiments, we report the execution time (Figure 5) and the entropy of the best solution found (Figure 6), as a function of the maximum number of candidates considered by our algorithm. We vary the number of items between 8 and 20, over the 150 domains of the `del.icio.us` dataset. In the graphs, we report the normalized values, averaged over all the 150 domains, as well as the standard deviation for these values (for most of the points standard deviation is too small and not visible). In order to make the results directly comparable to each other, we first normalize each series using the minimum (maximum) value of its regression line for the time (entropy) graph. Then, we compute the average normalized series, and its deviation.

In Figure 5, we report the averaged normalized execution times versus memory budget. We observe, that an increase in number of items results to an increase in complexity. Overall, the algorithm scales linearly with respect to the memory budget. When the number of items becomes large, the complexity is still determined by the memory budget (remember, that at each iteration the number of refinements is proportional to the size of the candidate set).

In Figure 6, we observe that for a small number of items, an increase in memory brings a considerably larger improvement in entropy, than for larger numbers of items. In the case of 8 items, the series drops until the entropy reaches its minimum for a maximum number of candidates of 32, which corresponds to the optimal solution. For larger number of items, the same effect is observed for a higher setting of the maximum number of candidates.

5.2. Synthetic dataset generation

In order to evaluate various properties of our approach in a controlled environment, we construct a synthetic dataset by generating itemsets for a number of dimensions closely resembling dimensions found in real datasets. These dimensions contain two to five itemsets of sizes up to three items, and we require exactly two dimensions to be present in each dataset. We demonstrate an example of generated dimensions in Table 3. Following the construction of dimensions, we use two different ways of assuring their distribution in data, which are represented in Algorithms 3 and 4 and described as follows:

Item-based generator (Algorithm 3) We calculate the frequency of singleton items by applying *Zipf's* law with a specified parameter z, $f(I_k) \sim 1/k^z$. We chose this distribution because it is known to resemble word frequencies in real-world datasets. Moreover, generating data from the Zipfian distribution allows to produce frequencies that are close to the uniform (when z is small), or

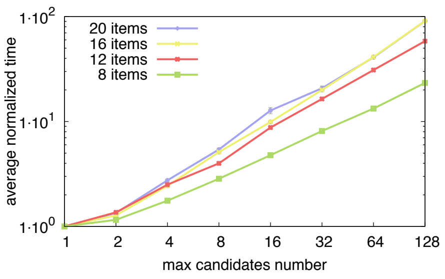

Fig. 5. Time vs. memory budget.

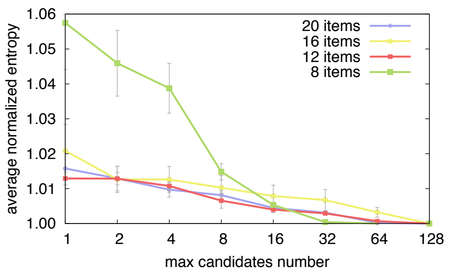

Fig. 6. Entropy vs. memory budget.

the exponential (when z is large) distributions. Both of the extreme cases are challenging for our problem, because with uniform frequencies non-dimensional items can occasionally form dimensions with a high coverage, and with exponential frequencies higher-order dimensions have a very small coverage.

We then add a uniform noise of level ν to these frequencies, and harmonize them for the items belonging to each of the dimension's itemsets (to account for the observed rule that itemsets in dimensions usually have equally frequent items; for example, both tags in {`eiffel` `tower`} usually appear together).

Coverage-based generator (Algorithm 4) We randomly pick a coverage for the itemsets of each dimension (i.e. the relative proportion of time each itemset represents its dimension), assuring that this value is not smaller than $1/|\delta^i|^2$. To all the items allocated to dimensions are assigned frequencies of their respecting itemsets. To the rest of the items are assigned frequencies according to *Zipf's* distribution, scaled to half of the minimal frequency of allocated items. This step is needed in order to assure that non-dimensional items would not occasionally form any dimensions at the time of generating itemsets. As in the previous method, we add a uniform noise of level ν to these frequencies.

Generating itemsets (Algorithm 5) For both of our methods, the itemsets are generated by iteratively sampling the distribution of items with respect to the specified dimensions. In this process, we use *Gibbs* sampling first to select a dimension (independently from other dimensions) and then to select the itemset representing it (allowing for co-occurrence with a level of 0.5ν). The rest of items, not covered by any dimension, are distributed with respect to their frequency.

In these experiments, we restrict the number of items to $n = 16$, which is equivalent to our minimum-support filtering on real datasets. Overall, we generate 10 thousand itemsets for each set of parameters for the first dataset and one thousand for the second one, to ensure a smooth distribution according to our model.

5.3. Experimental results on synthetic datasets

We assess the quality of the identified dimensions by comparing them to the dimensions used while generating the dataset. Our first similarity measure (used in Figure 7) is based on *Hamming* distance d between two dimensions (represented as binary vectors) divided by the total number of items: $\text{sim}(\Delta; \Delta_0) = 1 - d(\Delta; \Delta_0)/n$.

Since this measure is not able to distinguish incorrect positioning of items

Algorithm 3: Generating Item-based synthetic data.

```
   Input   : Dimensions $\Delta = \{\delta^i\}$, number of itemsets $n$, noise level $\nu$, Zipf $z$
   Output: Dataset $\mathcal{D}$ of itemsets
 1 //generate frequencies for singleton items i, using Zipf's law with noise
 2 $f[I_k] = (1 + \nu\xi)/k^z$ //random variable $\xi$ is evenly distributed on [-0.5;0.5]
 3 //normalize and regularize frequencies
 4 $f[I_k] = f[I_k]/\sum f[I_k]$;
   $\forall X^i \in \delta^i$ from $\Delta$ and $\{I^i_k, I^i_l\} \in X^i : |f[I^i_k] - f[I^i_l]| \leq \nu$;
 5 return $\mathcal{D}$ = GenerateItemsets($\Delta$,$n$,$\nu$,frequencies);
```

Algorithm 4: Generating Coverage-based synthetic data.

```
   Input   : Dimensions $\Delta = \{\delta^i\}$, number of itemsets $n$, noise level $\nu$,
             Zipf $z$, coverage $C$.
   Output: Dataset $\mathcal{D}$ of itemsets
 1 //randomly generate frequencies for dimension itemsets
 2 $f[X^i_j] = 1/|\delta^i| + (1 - 1/|\delta^i|)\xi_1$ //$\xi_1$ is uniformly distributed on [0;1]
 3 $f[X^i_j] = f[X^i_j]/f[\delta^i]$; //normalize frequencies within each dimension
 4 //add noise to frequencies
 5 forall the items $I_k$ do
 6     if $I_k \in X^i_j$ then  $f[I_k] = f[X^i_j]$;
 7     else  $f[I_k] = 0.5\,C$ min($f[X^i_j]$)$/k^z$;
 8     $f[I_k] = f[I_k](1 + \nu\xi_2)$; //$\xi_2$ is uniformly distributed on [-0.5;0.5]
 9 end
10 return $\mathcal{D}$ = GenerateItemsets($\Delta$,$n$,$\nu$,frequencies);
```

Algorithm 5: Generating itemsets GenerateItemsets$(\Delta, n, \nu, frequencies)$.

```
 1 forall the transaction $t \in \mathcal{D}$, $1..n$ do
 2     while $|t| = 0$ do
 3         forall the dimension $\delta^i \in \Delta$ do
 4             //sample the dimension using uniform $\xi_3$ and $\xi_4$
 5             if $\xi_3 < C$ then
 6                 repeat sample $X^i$ with co-occurrence level at $0.5\nu$
 7                     //sample the itemset using Gibbs method
 8                     $X^i$ = GibbsSampling($\delta^i, f[X^i]$); $t = t \cup X^i$;
 9                 until $\xi_4 > 0.5\nu$;
10             end
11         end
12         forall the non-allocated items $I_k$ do
13             //sample according to $f[I_k]$ using Gibbs method
14             $I_k$ = GibbsSampling($\mathcal{I}, f[I_k]$); $t = t \cup I_k$;
15         end
16     end
17 end
18 return $\mathcal{D}$;
```

Table 3. An example of dimensions used by Algorithms 3 and 4

Δ_1	{{0}, {1}, {2}, {3, 4}}, {{5, 6}, {7}}
Δ_2	{{0}, {1, 2}, {3, 4}}, {{5, 6}, {7}}
Δ_3	{{0}, {1}, {2}}, {{5}, {6}}
Δ_4	{{0}, {2}, {3}}, {{4, 5}, {6}, {7}}
Δ_5	{{0}, {1}, {2}, {3}, {4}}, {{5}, {6}, {7}}
Δ_6	{{0, 1}, {2}, {3}, {4}}, {{5}, {7}}
Δ_7	{{0}, {1}, {2, 3}, {4}}, {{6}, {7}}
Δ_8	{{0}, {1, 2}, {3, 4}}, {{5}, {6, 7}}
Δ_9	{{0}, {1}, {3}}, {{2}, {4}}
Δ_{10}	{{0}, {2}, {4}}, {{1}, {3, 5}}

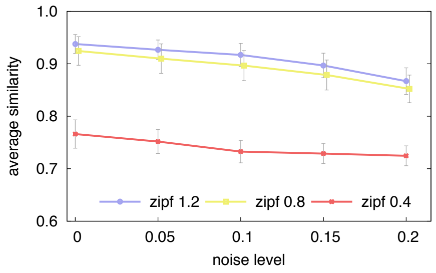

Fig. 7. Similarity to optimal dimensions versus noise.

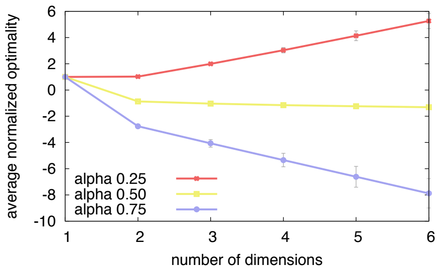

Fig. 8. $f(\Delta)$ dependency on the number of dimensions.

within identified dimensions, in the subsequent experiments we use its more elaborate derivative, i.e., averaging similarities between the corresponding itemsets within each dimension. To achieve this, we first map dimensions to each other according to maximal $\text{sim}(\Delta_i; \Delta_j)$ between them, and then average similarities between itemsets using the same mapping method.

These measures take values in the range $[0, 1]$, with higher values indicating stronger similarity: a value of 1 means that the algorithm correctly identified the planted dimensions. We note that these measures do not account for the varying significance of items, which is not favoring our approach, since including low-support items in the dimensions represents a challenge, even without the additional noise.

The evaluation of quality against noise for different parameters z is shown in Figure 7. In gray lines we plot the 0.95 confidence intervals for average values.

We can see that regardless of the noise added, our method is able to reconstruct almost perfectly the optimal dimensions for a wide range of distributions. As expected, the similarity between the identified and the optimal dimensions decreases on average with growing noise, and is significantly lower for smaller parameters z (more uniform items distribution).

In Figure 8 we evaluate the monotonicity of $f(\Delta)$ over the number of dimensions k, for different values of α parameter. It is clearly visible that for small values of α optimality gets higher (worse), while for large values every new dimension improves optimality (albeit, not the quality of extracted dimensions). For our experiments we chose $\alpha = 0.5$, since it provides a good balance between orthogonality and interestingness, and allows to rely on Theorem 2 (controlling the decomposition) for a wide range of data distributions.

Table 4. Top dimensions for different domains in `flickr` and `del.icio.us`.

δ^i	**collection of itemsets for** δ^i (`flickr`)
	domain **"jaguar"**
1	{`automobile`}, {`zoo`}
2	{`etype`}, {`auto`}
	domain **"eiffel tower"**
1	{`paris france europe tower`}, {`lasvegas`}
2	{`night seine`}, {`holiday travel`}
3	{`architecture`}
	domain **"pyramid"**
1	{`egypt giza cairo sphinx`}, {`louvre paris museum glass`}, {`mexico maya ruins`}, {`sanfrancisco transamerica`}
2	{`france sky`}, {`travel teotihuacan`}
3	{`architecture night`}, {`chichenitza`}
	domain **"hollywood"**
1	{`losangeles california sign`}, {`star film actor`}
2	{`us universalstudios`}, {`hollywoodboulevard night`}
3	{`theatre party sunset`}, {`canon street`}
	domain **"art"**
1	{`painting drawing`}, {`graffiti streetart`}, {`sculpture museum`}, {`newyork`}, {`color`}, {`photo`}, {`street`}
	domain **"spain"**
1	{`barcelona catalonia`}, {`madrid europe`}, {`andalusia granada`}, {`seville`}, {`valencia`}, {`holiday travel`}
2	{`architecture`}

δ^i	**collection of itemsets for** δ^i (`del.icio.us`)
	domain **"nytimes.com"**
1	{`news politics`}, {`food health`}, {`science`}, {`article`}, {`business`}, {`technology`}
	domain **"dpreview.com"**
1	{`photo camera review`}, {`dslr`}
2	{`digital`}
3	{`shopping`}
	domain **"lifehacker.com"**
1	{`howto lifehacks tips`},{`software windows tools freeware`}
2	{`firefox internet`}, {`linux utilities`}, {`email extensions`}, {`mp3 download`}, {`organization toread`}, {`photography`}
	domain **"apple.com"**
1	{`mac osx software`},{`ipod itunes music`},{`video quicktime`}, {`movies trailers`},{`iphone`},{`podcast podcasting`},{`technology`}
2	{`macosx howto`}
	domain **"microsoft.com"**
1	{`windows software tools`},{`.net programming`}
2	{`security xp`}
3	{`utilities`}
	domain **"ixbt.com"**
1	{`hardware software news computers russian`}, {`photo photography`}
2	{`article`}
3	{`reviews`}

5.4. Qualitative results on real datasets

We now report results on a qualitative evaluation of the proposed approach. We ran our algorithms on a set of different domains from `flickr`, `del.icio.us` and `dblp` datasets: "eiffel tower", "art", "hollywood", "pyramid", and "spain" for `flickr`; "nytimes.com", "lifehacker.com", "dpreview.com", "apple.com", "microsoft.com", and "ixbt.com" for `del.icio.us`; and "vldb", "cikm", "sigir" and "www" for `dblp`. We use a 3% minimum support threshold on items for all domains. The results of this experiment are summarized in Tables 4 and 5, where for each domain we report the top dimensions identified by our algorithm. We should note, that because of the fixed minimum support threshold, for some of the domains all available items are allocated to the first few dimensions, thus resulting in the varying number of dimensions being identified. In every case, we limit this number to the 3 top dimensions.

The dimensions reported by our algorithm are successfully describing the different concepts under each domain. For example, under the "eiffel tower" domain, we have as first dimension the *Eiffel Tower in Paris and Las Vegas,*[5] as second dimension *holidays in Paris*, and as third dimension *architecture*, all of which are different concepts related to "eiffel tower". Similarly, the "dpreview.com" domain in the del.icio.us dataset is described by the concepts of *photographic camera reviews*, *digital [photography]*, and *shopping*.

The results of this experiment demonstrate that our approach can effectively identify the diverse concepts related to some domain, in an automatic fashion.

[5] The city of Las Vegas (NV, USA) hosts a replica of the Eiffel Tower.

Table 5. Top dimensions for different conferences in `dblp` (by decades).

δ^i	**collection of itemsets for** δ^i
	conference **VLDB** ('80-'90)
1	{`database`}, {`data`}
2	{`queries`}, {`objects`}, {`transactions`}, {`algorithms`}, {`applications`}, {`large`}
3	{`interface`}, {`logic`}
	conference **VLDB** ('90-'00)
1	{`database`}, {`data`}
2	{`queries`}, {`objects`}, {`performance`}, {`integration`}, {`indexing`}, {`large`}
3	{`optimization`}, {`relations`}
	conference **VLDB** ('00-'10)
1	{`data`}, {`queries`}
2	{`database system`}, {`xml index`}, {`search`}, {`information`}, {`network`}
3	{`optimization`}, {`view`}
	conference **WWW** ('00-'05)
1	{`web`}, {`search engine`}
2	{`model`}, {`approach`}, {`content`}
3	{`application`}, {`page`}, {`xml`}
	conference **WWW** ('06-'10)
1	{`web`}, {`social network`}
2	{`user`}, {`engine`}, {`approach`}
3	{`service`}, {`page`}, {`framework`}

δ^i	**collection of itemsets for** δ^i
	conference **SIGIR** ('80-'90)
1	{`retrieval information`}, {`system`}
2	{`documents`}, {`text`}
3	{`data`}, {`approach`}
	conference **SIGIR** ('90-'00)
1	{`retrieval`}, {`search`}
2	{`information`}, {`documents`}
3	{`text`}, {`queries`}
	conference **SIGIR** ('00-'10)
1	{`retrieval`}, {`search`}
2	{`answer prediction`}, {`summarization`}
3	{`filtering results`}, {`ir`}
	conference **CIKM** ('90-'00)
1	{`database`}, {`information`}
2	{`queries`}, {`data`}, {`index`}, {`models`}, {`objects`}, {`approach`}, {`management`}
3	{`association`}, {`classification`}
	conference **CIKM** ('00-'10)
1	{`queries`}, {`data`}
2	{`search`}, {`information`}, {`retrieval`}, {`system`}, {`text`}, {`web`}, {`documents`}, {`models`}
3	{`extraction`}, {`analysis`}, {`detection`}

Finally, we observe that our algorithm provides meaningful results, even when operating on noisy tagsets, such as `flickr` and `del.icio.us`, which contain a large number of non-useful tags.

To counter the results shown on tagset annotations, we additionally evaluated our algorithm on a different kind of data: publication titles of papers in different database and information retrieval conferences from the `dblp` dataset. The main difference between this dataset and `flickr` and `del.icio.us` is that keywords are often used in combination (in order to express a specific notion or idea), and have larger variability than tags. Moreover, noun words and verbs have different distributions across titles, making this dataset harder to process for our method.

In this set of experiments, we apply our method in order to process the paper titles of various conferences separately, and over different time intervals. This analysis can show us not only what the prevalent dimensions for each conference are and how they compare to other conferences, but also how these dimensions evolve over time. We report the discovered dimensions during different time intervals in Table 5.

In this case, dimensions appear as mixed sets of scientific topics and relevant methods, mainly due to their conditional dependency (in the paper titles). Still, it is possible to compare topics of conferences and see their evolution over time. For instance, the 3rd dimension for VLDB demonstrates that topics have changed from "interface" and "logic" to "relations" and "optimization" and finally to "view" and "optimization" in the past three decades. Another example of topic change is visible in WWW, where the initial dominance of "search engine" has been replaced by "social network".

Table 6. Various itemset similarity measures.

Measure	Formula	Description
Binary Inclusion	$\begin{cases} 1, & X \subseteq t; \\ 0, & X \nsubseteq t. \end{cases}$	The most strict similarity measure, which requires all of the items to be present in the transaction
Cosine Similarity	$\frac{\lvert X \cap t \rvert}{\lVert X \rVert \, \lVert t \rVert}$	A less strict measure, which, however, performs less well for long transactions
Jaccard Coefficient	$\frac{\lvert X \cap t \rvert}{\lvert X \cup t \rvert}$	A more sensitive measure, which after the normalization provides almost length-independent probabilities
Matched Fraction	$\frac{\lvert X \cap t \rvert}{\lvert X \rvert}$	A non-symmetric measure independent of transaction's length, which is good to handle items ambiguity
Weighted Fraction	$\frac{\sum_{X \cap t} \log^{-1}(k+1)}{\sum_{k=1}^{n} \log^{-1}(k+1)}$	A measure similar to Matched Fraction, but items are weighted according to their relative supports

5.5. Probability measures

Following our qualitative evaluation, we now study and compare the different probability measures, which determine the characteristics of the identified dimensions. This becomes particularly relevant in the context of our optimization problem, where a probability space affects the overall shape of the optimization objective and its convergence.

As we point out in Section 3.1, our approach is generic and can assume various probability measures. Probability similarity measures, different than the binary, may become useful for domains where itemsets can be approximate or uncertain, thus requiring less strict, or probabilistic matching between itemsets.

In this study, we explore the use of five such measures and mention their specific properties (summarized in Table 6). Additionally, we provide a few intuitions, which we followed when designing them: (*i*) a probability measure can treat patterns X and database transactions t differently, according to their semantics; (*ii*) probabilities should not be affected by a varying length of t after the normalization; (*iii*) we can weight the occurrences of items from X in t, according to their importance; (*iv*) very small probabilities can make entropy's penalty of a non-coverage or co-occurrence ineffective. We discuss these intuitions in a more detailed way on the concrete examples below.

Binary Inclusion facilitates the selection of concise itemsets with high coverage. It is sensitive to mutual exclusivity of itemsets, leading to compact and noiseless dimensions. This similarity measure works best for datasets with low ambiguity among items. For example, if there co-exist multiple spellings of the same tag (usually non co-occurring), binary inclusion will treat them as separate itemsets in a dimension, or even as itemsets from another dimension.

Cosine Similarity is an approximate measure, dependent both on the lengths of a pattern and an itemset. Therefore, it tends to minimize the co-occurrence by refining patterns (through addition of items). This process may lead patterns in a dimension to be unusually long and with low coverage.

Jaccard Coefficient shares similar properties to cosine similarity, except that it is more sensitive to the overlap between a pattern and an itemset.

Matched Fraction handles the ambiguity of items better than the other measures, but it may allow patterns with low-support items, if they do not co-occur with the other patterns in a dimension.

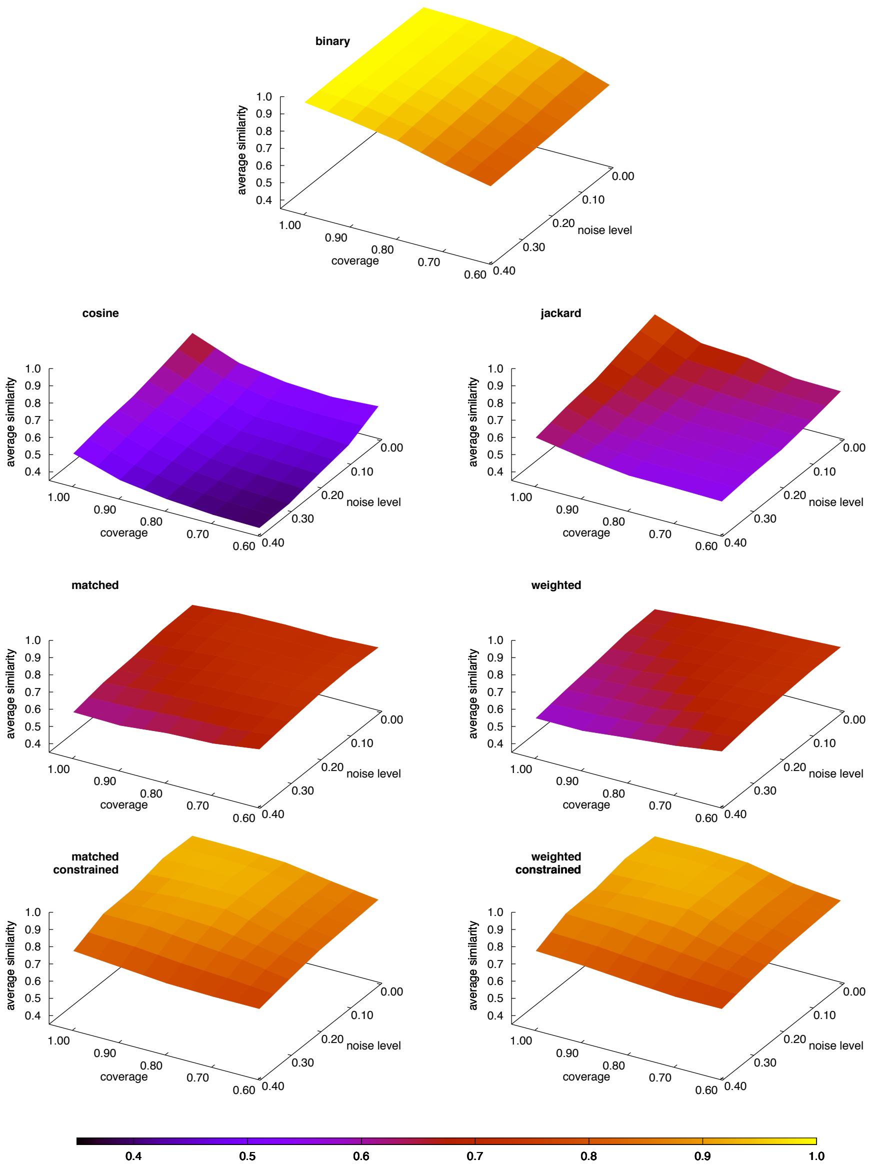

Fig. 9. Similarity to optimal dimensions versus noise and coverage.

Weighted Fraction is similar to matched fraction, but it weights overlapping items according to their importance (frequency), thus approximating (but not substituting) a coverage-based pruning behavior.

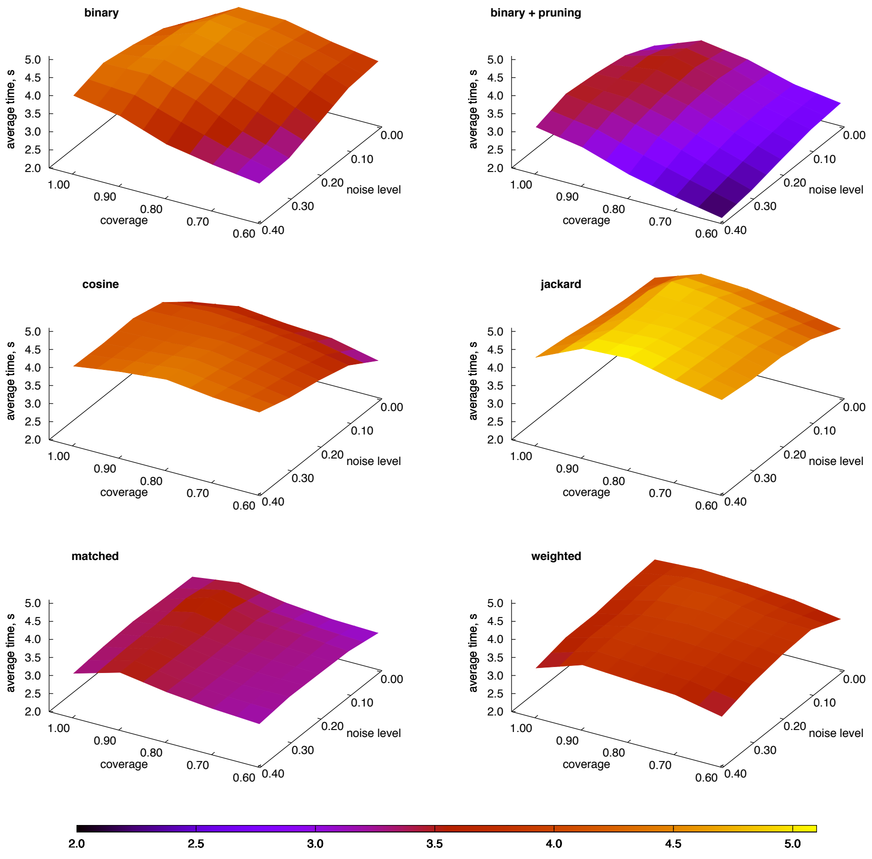

Fig. 10. Average time to discover dimensions versus noise and coverage.

5.6. Evaluation of probability measures

We evaluated the general performance of these measures on a collection of synthetic datasets generated by Algorithm 4, as well as on the real datasets.

First, we generated a database of 200 dimensions templates, each containing two dimensions. Then, for each of the parameter combinations (coverage, noise level) and each of these templates we generated a dataset of 1000 itemsets. We varied coverage from 1.0 to 0.6 and noise level from 0.0 to 0.4 using 0.1 increments. Since all the 200 generated datasets use the same set of dimensions templates, it is possible to measure and compare properties for various parameter combinations. In our evaluation, we run each experiment on all datasets, and report the mean values. We omit the presentation of confidence intervals for brevity, but we note that all results are concentrated close to the corresponding mean values. We also note that all the observed dependencies are statistically significant.

In Figure 9 we evaluate the quality of the identified dimensions by measuring their similarity to the optimal dimensions (used for constructing the dataset). Binary similarity demonstrates very predictable performance, having about 100%

precision in the case of 100% coverage, and then gradually loosing precision as the parameters become less favorable. The performance is still acceptable, at 75%, when we have high noise and low coverage. It is worth mentioning, that the average precision is exactly 100% (0% variance) in the case of 30% noise, while this is not true for 0% noise, as one might expect. This behavior can be explained by the fact that our method does not include the length of itemsets into the optimization criterion, and dimensions with smaller number of items and same coverage are indistinguishable. On the other hand, noise helps to differentiate the coverage for various itemset refinements, and facilitates selecting optimal and complete dimensions.

Cosine and Jaccard similarities show more dramatic decreases in performance. Moreover, the quality of identified dimensions is less than 100% without noise, indicating that these measures are not suitable for itemset distributions used in our datasets.

Matched and Weighted similarities perform more stable across the parameters ranges (the similarity to optimal dimensions is always between 0.55-0.75), yet they demonstrate a surprising behavior: precision grows with decreasing coverage. This behavior, which is statistically significant, can be explained as follows.

- Minimizing entropy forces selecting very short (or very long) itemsets which maximize (or minimize) the probabilities;
- Both kinds of itemsets are very sensitive to co-occurrence;
- Co-occurrence decreases proportionally with the coverage, while
- Short itemsets have virtually the same coverage as the optimal ones and long itemsets have even larger overall coverage (not in a binary sense), which decreases slower than the actual binary coverage (used as a parameter);
- The optimality of longer itemsets grows when the difference between coverage and co-occurrence becomes more pronounced, with the decreasing coverage.

The above conditions result in selecting itemsets of proper length with the decreasing coverage.

When we couple these similarity measures with the constraint based on minimal coverage, we observe that quality improves (refer to the bottom graphs in Figure 9). The results show that we can achieve a better quality for high coverage and low noise (up to 0.93). Quality for low coverage and high noise settings drops to 0.75, but remains always better than without constraining the itemsets. Note that in this case, we use Lemma 2 in order to stop refining the itemsets. Without this constraint, longer itemsets will be preferred due to the nature of applied probability measure, leading to solutions with less frequent and more noisy itemsets. Even though Lemma 2 was proven for binary probabilities, we can still use it here as a constraint, because it is based on the entropy measure, which is still indicative of the information content of dimensions (that is, coverage) even when different probability measures are applied. Note that instead of Lemma 2, we could use any hard constraint on coverage, in order to achieve a similar behavior.

Figure 10 depicts the dependency of execution time to the same parameters of coverage and noise (the reported results are again averages over 200 experiments). We observe that for binary similarity, the execution time depends inversely on noise, and decreases along with coverage. Such improvement of execution time, when the dataset quality decreases, has a simple explanation: noise-free and good-coverage dimensions require more iterations to converge, since till the end of the optimization process multiple revisions of optimal dimensions are being concurrently optimized. On the other hand, the suboptimal dimensions in noisy data move out of the allocated memory budget quite fast, and consequently, the algorithm converges faster. However, this does not imply that the identified dimensions are better, as can be verified with the help of Figure 9.

As can be seen in Figure 10 (top right), our pruning technique based on Lemma 2 demonstrates a considerable improvement of execution time for the binary method (25% on average).

In terms of performance, matched and weighted similarities (refer to the bottom graphs of Figure 10) are faster than the rest, since in this case a memory budget is very quickly flooded with various sub-optimal refinements, which can not lead to any optimality improvements, therefore, preventing further iterations.

6. Conclusions and future work

Motivated by applications on repositories of annotated resources in the collaborative tagging domain, we introduce the problem of diverse dimension decomposition in transactional databases. In particular, we adopt an information-theoretic perspective on the problem, relying on entropy for defining a single objective function that simultaneously captures constraints on coverage, exclusivity and orthogonality.

We present an approximate greedy method for extracting diverse dimensions, that exploits the FP-tree representation of the input transactional dataset and clever pruning techniques. Our experiments on datasets of tagged resources from `flickr` and `del.icio.us` confirm effectiveness and efficiency of our proposal, and analysis of titles from `dblp` demonstrates a possibility of applying diverse dimension decomposition to text datasets as well. The assessment on synthetic and artificially noisy data confirms that our method is able to reconstruct the "true" dimensions, and it withstands noise.

In our future investigations, we plan to have a user study for evaluating the discovered dimensions in different domains. A possibility is also that of developing a vertical application exploiting our method for mining diverse dimensions in order to detect, in unsupervised and automatic fashion, collection of web sites with diverse content from `del.icio.us`.

Acknowledgements. Francesco Bonchi and Aristides Gionis were partially supported by the Torres Quevedo Program of the Spanish Ministry of Science and Innovation, co-funded by the European Social Fund, and by the Spanish Centre for the Development of Industrial Technology under the CENIT program, project CEN-20101037, "Social Media" (www.cenitsocialmedia.es).

References

[1] A. J. Knobbe and E. K. Y. Ho, "Maximally informative k-itemsets and their efficient discovery," in *KDD*, T. Eliassi-Rad, L. H. Ungar, M. Craven, and D. Gunopulos, Eds. ACM, 2006, pp. 237–244.

[2] ——, "Pattern teams," in *PKDD*, ser. Lecture Notes in Computer Science, J. Fürnkranz, T. Scheffer, and M. Spiliopoulou, Eds., vol. 4213. Springer, 2006, pp. 577–584.

[3] N. Tatti, "Maximum entropy based significance of itemsets," in *Knowledge and Information Systems (KAIS)*, vol. 17. Springer, 2008, pp. 57–77.

[4] ——, "Probably the best itemsets," in *KDD*, B. Rao, B. Krishnapuram, A. Tomkins, and Q. Yang, Eds. ACM, 2010, pp. 293–302.
[5] J. V. Michael Mampaey, Nikolaj Tatti, "Tell me what i need to know: succinctly summarizing data with itemsets," in *KDD*, 2011.
[6] H. Heikinheimo, E. Hinkkanen, H. Mannila, T. Mielikäinen, and J. K. Seppänen, "Finding low-entropy sets and trees from binary data," in *KDD*, 2007.
[7] C. Zhang and F. Masseglia, "Discovering highly informative feature sets from data streams," in *DEXA*, 2010.
[8] J. Han, J. Pei, and Y. Yin, "Mining frequent patterns without candidate generation," in *ACM SIGMOD Conference*, 2000, pp. 1–12.
[9] M. Tsytsarau, F. Bonchi, A. Gionis, and T. Palpanas, "Diverse Dimension Decomposition of an Itemsets Space," in *ICDM*, 2011.
[10] B. Sigurbjörnsson and R. van Zwol, "Flickr tag recommendation based on collective knowledge," in *WWW*, 2008.
[11] R. van Zwol, B. Sigurbjörnsson, R. Adapala, L. G. Pueyo, A. Katiyar, K. Kurapati, M. Muralidharan, S. Muthu, V. Murdock, P. Ng, A. Ramani, A. Sahai, S. T. Sathish, H. Vasudev, and U. Vuyyuru, "Faceted exploration of image search results," in *WWW*, 2010.
[12] M. Grahl, A. Hotho, and G. Stumme, "Conceptual clustering of social bookmarking sites," in *LWA 2007: Lernen - Wissen - Adaption*, 2007.
[13] D. Ramage, P. Heymann, C. D. Manning, and H. Garcia-Molina, "Clustering the tagged web," in *WSDM 2009: Proc. of the 2nd ACM Int. Conference on Web Search and Data Mining*, 2009.
[14] M. van Leeuwen, F. Bonchi, B. Sigurbjörnsson, and A. Siebes, "Compressing tags to find interesting media groups," in *CIKM*, 2009.
[15] K. Morik, A. Kaspari, M. Wurst, and M. Skirzynski, "Multi-objective frequent termset clustering," in *Knowledge and Information Systems (KAIS)*, vol. 30. Springer, 2012, pp. 715–738.
[16] F. Bonchi, C. Castillo, D. Donato, and A. Gionis, "Topical query decomposition," in *KDD*, 2008.
[17] B. Carterette and P. Chandar, "Probabilistic models of ranking novel documents for faceted topic retrieval," in *CIKM*, 2009.
[18] R. L. Santos, C. Macdonald, and I. Ounis, "Exploiting query reformulations for web search result diversification," in *WWW*, 2010.
[19] G. Capannini, F. M. Nardini, R. Perego, and F. Silvestri, "Efficient diversification of web search results," *PVLDB*, vol. 4, no. 7, pp. 451–459, 2011.
[20] F. Korn, A. Labrinidis, Y. Kotidis, and C. Faloutsos, "Quantifiable data mining using ratio rules," *VLDB J.*, vol. 8, no. 3-4, pp. 254–266, 2000.
[21] G. H. Golub and C. F. Van Loan, *Matrix Computations*, 3rd ed. The Johns Hopkins University Press, October 1996.
[22] F. Verhein and S. Chawla, "Geometrically inspired itemset mining," in *ICDM*, 2006, pp. 655–666.
[23] T. M. Cover and J. A. Thomas, *Elements of Information Theory.* New York, NY, USA: Wiley & Sons, 1991.

Author Biographies

Mikalai Tsytsarau is a PhD candidate at the Database and Information Management Group dbTrento of the department of Information Engineering and Computer Science DISI, at the University of Trento, working with professor Themis Palpanas. He received his Master's in Computer Science from the Belarusian State University and worked as a software engineer at Epam Systems. Mikalai also worked at Yahoo! Research and HP Labs as a visiting researcher and at Qatar Computing Research Institute as research associate. Mikalai's main scientific interests are in Data Management and Information Retrieval, with a focus on the web environment. The list of his current research activities includes: large-scale Opinion Mining, Itemset Mining, Databases and Indexing structures. He also has an interest in Click-log Mining and Entity Re-ranking for search engines.

Francesco Bonchi is a senior research scientist at Yahoo! Research in Barcelona, Spain, where he is part of the Web Mining Group. His recent research interests include mining query-logs, social networks, and social media, as well as the privacy issues related to mining these kinds of sensible data. In particular, his main interest nowadays is to develop data mining methods for the analysis of information an influence spread. He gave a keynote talk at WI-IAT 2011 on this topic. He has been program co-chair of the European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases (ECML PKDD 2010), and various workshops on social web mining, privacy and other topics in data mining.

Aristides Gionis is a senior research scientist in Yahoo! Research, Barcelona. He received his Ph.D from the Computer Science department of Stanford University in 2003 and until 2006 he has been with the University of Helsinki. He is serving on the editorial boards of TKDE and KAIS and he has been in the program committee of numerous premier conferences. His research interests include algorithms for data analysis and applications in the web domain.

Themis Palpanas is a professor of computer science at the University of Trento, Italy. He received his PhD degree from the University of Toronto, Canada. Before joining the University of Trento, he worked at the IBM T.J. Watson Research Center. He has also worked for the University of California, Riverside, and visited Microsoft Research and the IBM Almaden Research Center. He is the author of five US patents, three of which are part of commercial products. He has received three Best Paper awards (PERCOM 2012, ICDE 2010 and ADAPTIVE 2009). He is serving on the Editorial Advisory Board of the Information Systems Journal and as an Associate Editor in the Journal of Intelligent Data Analysis. He is General Chair for VLDB 2013, has served on the program committees of several top database and data mining conferences.

Correspondence and offprint requests to: Mikalai Tsytsarau, via Sommarive 14, Povo (TN) 38123, Italy. Email: tsytsarau@disi.unitn.it