Comparing Apples and Oranges
Measuring Differences between Data Mining Results

Nikolaj Tatti and Jilles Vreeken
Advanced Database Research and Modeling
Universiteit Antwerpen
{nikolaj.tatti,jilles.vreeken}@ua.ac.be

Abstract. Deciding whether the results of two different mining algorithms provide significantly different information is an important open problem in exploratory data mining. Whether the goal is to select the most informative result for analysis, or decide which mining approach will likely provide the most novel insight, it is essential that we can tell how different the information is that two results provide.
In this paper we take a first step towards comparing exploratory results on binary data. We propose to meaningfully convert results into sets of noisy tiles, and compare between these sets by Maximum Entropy modelling and Kullback-Leibler divergence. The measure we construct this way is flexible, and allows us to naturally include background knowledge, such that differences in results can be measured from the perspective of what a user already knows. Furthermore, adding to its interpretability, it coincides with Jaccard dissimilarity when we only consider exact tiles.
Our approach provides a means to study and tell differences between results of different data mining methods. As an application, we show that it can also be used to identify which parts of results best redescribe other results. Experimental evaluation shows our measure gives meaningful results, correctly identifies methods that are similar in nature, and automatically provides sound redescriptions of results.

1 Introduction
Deciding whether the results of different mining algorithms provide significantly different information is an important, yet understudied, open problem in exploratory data mining. Whether we want to select the most promising result for analysis by an expert, or decide which mining approach we should apply next in order to most likely gain most novel insight, we need to be able to tell how different the information is that different results, by possibly different methods, provide. However, while the comparison of results is a well-studied topic in statistics, it has received much less attention in the knowledge discovery community.
Clearly, any dataset only contains a limited amount of knowledge—which is the most that we can hope to discover from it. To extract this information, we have an ever growing number of data mining algorithms at our disposal. However, most data mining results are complex, and their analysis and validation often
takes considerable effort and cost. So, simply applying ‘all’ methods and letting
an expert analyse ‘all’ results is not a feasible approach to extract ‘all’ knowledge.
Moreover, many of these results will be redundant, i.e. convey roughly the same
information, and hence only require effort while not providing extra insight.

Instead, we would ideally just select that result for analysis which will provide
us the most new knowledge. In order to be able to do this, two basic requirements
have to be met. First of all, we need to be able to measure how different two
results are from an information-providing perspective; if they essentially provide
the same information, we could just select one for processing. Second, we should
be able to include our background knowledge, such that we can gauge the amount
of information a result gives us compared to what we already know.

Although an important practical problem, it has been surprisingly under-
studied in data mining. The main focus in exploratory data mining research has
mostly been on developing techniques to discover structure, and, not so much on
how to compare between results of different methods. As a result there currently
exist no general methods or theory to this end in the data mining literature.

For tasks where a formal objective is available, we can straightforwardly use
it to compare fairly between the results of different methods. In classification,
for instance, we can use accuracy. For exploratory data mining, however, there
is no formal common goal: any result that provides novel insight is potentially
useful. The core of the problem is thus that comparing between methods is
like comparing apples to oranges: a clustering is a different result than a set of
itemsets, which are, in turn, different from a classifier, set of subgroups, etc. So,
in order to make a sensible comparison, we need to find a common language.

In this regard, the comparison of complex objects, e.g. of datasets [23,25], is
related. Our setting, however, is more general, as now we do not want to compare
between one type of complex object, but want to consider a very rich class of
objects—potentially consisting of any data mining result. Arguably, some of the
most general complex objects to compare between are probability distributions.
Statistics and Information Theory provide us tools for measuring differences
between distributions, such as Kullback-Leibler divergence [2]. Mining results,
however, rarely are probability distributions, and if they are, not necessarily for
the same random variable; making these tools unsuited for direct application.

A simple yet important observation we make is that any mining result essen-
tially identifies some properties of the dataset at hand. In an abstract way, we
could identify all datasets for which these properties hold. This is an important
notion, as it provides us a way to compare between results of different methods:
if two results provide the same information, they identify the same subspace of
possible datasets, and the more different the information two results give, the
smaller the overlap between the sets of possible datasets will be. More generally
put: every data mining result implicitly defines a probability distribution over
datasets. And hence, if we can model these distributions, we can use standard
tools from Statistics to compare between data mining results fair and square.

In this paper, we propose to translate data mining results into probability dis-
tributions over datasets using the Maximum Entropy principle [3]. It allows us to
uniquely identify the model that makes optimal use of the provided information, but is fully unbiased otherwise. By subsequently measuring the Kullback-Leibler divergence between these distributions, we can tell how different two results are from an information perspective. Besides data mining results, we can also incorporate background knowledge into our model, and so use it to score results from specific points of view [5].

Finding these probability distributions, and conditioning them using data mining results, however, is far from trivial. Here, we therefore give a proof of concept of our approach for binary data, for which the basics of maximum entropy modelling are available. We show that many exploratory data mining results on binary data can easily be translated into sets of noisy tiles: combinations of rows and columns, for which we know the density of 1s. We show we can efficiently acquire the maximum entropy distribution given sets of such tiles, and that by KL divergence we can so compare between results.

More specifically, in our experiments we compare between the results of ten different exploratory data mining methods, including (bi-)clusters, subspace clusters, sets of tiles, and sets of frequent itemsets. Moreover, we give a theoretic framework to mine for redescriptions. That is, given a (sub)set of noisy tiles from one result, we can identify the set of tiles from another result that best approximates the same information. Experiments show our measure works well in practice: dissimilarity converges to 0 when models approximate each other, methodologically close methods are correctly grouped together, and sensible redescriptions for tile-sets are obtained. In other words, we give an approach by which we can meaningfully mix apples and oranges, and compare them fairly.

The roadmap of this paper is as follows. Next, in Section 2 we give the notation and preliminaries we use throughout the paper. Section 3 details how we can build a global model from a set of tiles, which we use in Section 4 to define a measure to compare such sets. In Section 5 we subsequently use this measure for redescribing sets of tiles. We discuss related work in Section 6, and we evaluate our measure empirically in Section 7. We round up with discussion and conclusions in Sections 8 and 9. Due to lack of space, we give the proofs, such as for NP-completeness, in the Appendix [24].

2 Preliminaries

In this section, we define the preliminaries we will use in subsequent sections.

A binary dataset \( D \) is a binary matrix of size \( N \times M \) consisting of \( N \) rows, binary vectors of size \( M \). We denote \((i,j)^{th}\) entry of \( D \) by \( D(i,j) \). We denote the space of all binary datasets of size \( N \times M \) by \( \mathcal{D} \).

We approach the comparison of different data mining results by first translating these into sets of tiles. A tile \( T = (t(T), a(T)) \) is a tuple consisting of two lists. The first list, \( t(T) \), is a set of integers between 1 and \( N \) representing the transactions of \( T \). The second list, \( a(T) \), is a set of integers between 1 and \( M \) representing the attributes. We define \( s(T) \) to be the cartesian product of \( t(T) \)
and \( a(T), s(T) = \{(i, j) \mid i \in t(T), j \in a(T)\} \). Given a tile set \( T \) we also define \( s(T) = \bigcup_{T \in T} s(T) \).

Given a tile \( T \) and a dataset \( D \) we define a frequency \( fr(T; D) \) to be the proportion of ones in \( D \) corresponding to the entries identified by \( T \),

\[
fr(T; D) = \frac{1}{|s(T)|} \sum_{i \in t(T), j \in a(T)} D(i, j).
\]

There are numerous techniques for mining tile sets but we can also naturally describe a large number of statistics and mining results using tile sets:

- **Density**: the frequency of a tile containing the whole data is equal to the density of the data.
- **Margins**: the frequency of a column \( i \) can be expressed with a single (noisy) tile containing the column \( i \) and all transactions. Analogously, we can express the margins for each row.
- **Itemsets**: any itemset can be converted into a tile by taking the supporting transactions. Thus, an itemset collection can be converted into a tile set.
- **Bi/Subspace-clustering**: subspace and bi-clusters are sets of transactions and columns. Hence, we can naturally represent these results by equivalent tiles.
- **Clustering**: Given a clustering, either over transactions or items, we can construct a tile set in two different ways. The first way is to represent each cluster by a single tile, representing the density of a tile. The other way is to compute (column) margins for each cluster and represent these margins by tiles. This is particularly natural for \( k \)-means, since a centroid then corresponds to the column margins of the corresponding transactions.

Let \( p \) be a distribution defined over \( D \), the space of all datasets of size \( N \times M \).

We define the frequency of a tile to be the average frequency with respect to \( p \),

\[
fr(T; p) = \sum_{D \in D} p(D)fr(T; D).
\]

We can also express the frequency directly by this distribution.

**Lemma 1.** Given a distribution \( p \) and a tile \( T \), the frequency is equal to

\[
fr(T; p) = \sum_{(i, j) \in s(T)} p((i, j) = 1),
\]

where \( p((i, j) = 1) \) is the probability of a dataset having 1 as \((i, j)\)th entry.

We say a tile is **exact** if its frequency is 0 or 1, and otherwise say it is **noisy**.

**Corollary 1.** For an exact tile \( T \), \( p((i, j) = 1) = fr(T; p) \), where \((i, j) \in s(T)\).

**Example 1.** Consider a dataset \( D \) given in Figure 1(a). We consider five different tiles, \( T_1 = (2, \ldots, 5) \times (1, \ldots, 5), T_2 = (1, 2) \times (1, 2), T_3 = (3, 4, 5) \times (1, 2), T_4 = (4, 5) \times (3, 4, 5), \) and \( T_5 = (3, 4, 5) \times (4, 5) \). The frequencies are \( fr(T_1; D) = 10/20 = 1/2, fr(T_2; D) = fr(T_4; D) = fr(T_5; D) = 1, \) and \( fr(T_3; D) = 0. \) By definition, \( T_2, \ldots, T_5 \) are exact, while \( T_1 \) is not.
Given two distributions, say $p$ and $q$, we resp. define entropy and Kullback-Leibler divergence as

$$H(p) = -\sum_{D \in \mathcal{D}} p(D) \log p(D) \quad \text{and} \quad KL(p \parallel q) = \sum_{D \in \mathcal{D}} p(D) \log \frac{p(D)}{q(D)}.$$  

### 3 Building Global Models from Tiles

To meet our goal, we have to construct a statistically sound technique for comparing two sets of tiles. In this section we construct a global model for datasets using the given tiles. We will use these models for comparing the tile sets.

Consider that we are given a tile set $\mathcal{T}$, and for each tile $T \in \mathcal{T}$ we are also given a frequency $\alpha_T$. Typically, the frequencies are obtained from the data at hand, $\alpha_T = fr(T; D_{in})$, but this is not a necessary condition. The tiles convey local information about the data $D_{in}$ and our goal is to infer a distribution $p$ over $\mathcal{D}$, that is, how probable data set $D \in \mathcal{D}$ is given a tile set $\mathcal{T}$. If the information at hand defines the data set uniquely, then $p(D) = 1$ if and only if $D = D_{in}$.

To derive the model, we use a well-founded notion from information theory, the Maximum Entropy principle [2]. Roughly speaking, by Maximum Entropy, we incorporate the given information into a distribution, yet further making it as evenly spread as possible. To define the distribution, we first define the space of distribution candidates. That is, the space of those distributions that produce the same frequencies for the given tiles, $\mathcal{P} = \{p \mid fr(T; p) = \alpha_T, \text{ for all } T \in \mathcal{T}\}$. In other words, $\mathcal{P}$ contains all distributions that explain the frequencies $\alpha_T$. From this set, we select one distribution, which we denote by $p^*_T$, such that $p^*_T$ maximises the entropy, $H(p^*_T) \geq H(p)$ for any $p \in \mathcal{P}$.

We will abuse notation and write $H(\mathcal{T})$ where we mean $H(p^*_T)$. Similarly we write $KL(\mathcal{T} \parallel \mathcal{U})$ to mean $KL(p^*_T \parallel p^*_U)$, where $\mathcal{U}$ is another tile set.

A classic theorem states that $p^*$ can be written as an exponential form.

**Theorem 1 (Theorem 3.1 in [3])**. Given a tile set $\mathcal{T}$, a distribution $p^*$ is the maximum entropy distribution if and only if it can be written as

$$p^*(D) \propto \begin{cases} \exp(\sum_{T \in \mathcal{T}} \lambda_T fr(T; D)) & D \notin \mathcal{Z} \\ 0 & D \in \mathcal{Z}, \end{cases}$$  

---

**Fig. 1.** Toy example of a dataset and several maximum entropy models.
Algorithm 1: Iterative Scaling for solving the MaxEnt distribution

```
input : tile set T, target frequencies {α_T}
output : Maximum entropy distribution p
1 p ← a matrix of size N × M with values 1/2;
2 foreach T ∈ T, α_T = 0, 1 do p(i, j) ← α_T for all (i, j) ∈ s(T);
3 while not converged do
4     foreach T ∈ T, 0 < α_T < 1 do
5         f ← fr(T; p);
6         x ← (α_T(1 - f))/(f(1 - α_T));
7         p(i, j) ← p(i, j)x/(1 - p(i, j)(1 - x)) for all (i, j) ∈ s(T);
```

where λ_T is a certain weight for fr(T; D) and Z is a collections of datasets such that p(D) = 0 for each p ∈ P.

The next theorem allows to factorize the distribution p* into a product of Bernoulli random variables, each variable representing a single entry in the dataset. Such a representation gives us a practical way for inferring the model.

**Theorem 2.** Let T be a tile set. Write T(i, j) = {T ∈ T | (i, j) ∈ s(T)} to be the subset of T containing the tiles that cover an entry (i, j). Then, the maximum entropy distribution can be factorized as p*(D) = ∏_{i,j} p*(i, j) = D(i, j), where

\[
p^*(i, j) = \frac{\exp\left(\sum_{T \in T(i, j)} \lambda_T\right)}{\exp\left(\sum_{T \in T(i, j)} \lambda_T\right) + 1} \quad \text{or} \quad p^*(i, j) = 1 = 0, 1 .
\]

Theorem 2 allows to represent p* as Bernoulli variables. We should stress that this is a different model than assuming independence between items in a random transaction. Our next step is to discover the correct frequencies for these variables. Here we use a variant of a well-known Iterative Scaling algorithm [4]. The algorithm is given in Algorithm 1. Informally said, given a tile T the algorithm updates the probabilities such that the frequency of T is closer to α_T. This is performed for each tile. Updating a single tile might change the frequency of another tile. Hence, we need several passes. A single pass takes O(NM) time. The original proof of correctness for iterative scaling assumes that p* has no zero probabilities. This is often violated in our setup. Hence we provide a proof of correctness in the Appendix.

**Theorem 3.** The iterative scaling algorithm given in Algorithm 1 correctly converges to the maximum entropy distribution.

It turns out, that if the given tiles are exact, the maximum entropy distribution is simple. A Bernoulli variable corresponding to the (i, j) entry is always 1 (or 0), if the entry is covered by an exact tile, i.e. with frequency 1 (or 0). Otherwise, the variable is equal to a fair coin toss. This form will allow us to express distances between sets of exact tiles in the next section.
Theorem 4. Let $T$ be a collection of exact tiles and let $\alpha_T$ be the desired frequency of a tile $T \in T$. Then

$$p^*_T((i, j) = 1) = \begin{cases} \alpha_T & \text{if there exists } T \in T \text{ such that } (i, j) \in s(T) \\ 1/2 & \text{otherwise} \end{cases}.$$ 

Example 2. Let us continue Example 1. Consider the following three tile sets $T = \{T_2, T_4\}$, $U = \{T_2, T_3, T_5\}$, and $B = \{T_1\}$. The corresponding maximum entropy models are given in Figure 1, such that each entry represents the probability $p^*((i, j) = 1)$. As the sets $T$ and $U$ contain only exact tiles, by Theorem 4 the entries for the models $p^*_T$, $p^*_U$, and $p^*_{T \cup U}$ are either 0, 1, or $1/2$.

Consider $p^*_{T \cup B}$. Corollary 1 states that entries in $s(T_2)$ and $s(T_4)$ should be 1, since both tiles are exact. From $T_1$, we know that there are 10 ones, yet $T_2$ and $T_4$ account only for 8. Hence, there should be 2 ones in the 12 entries outside of $T$, on average. We aim to be as fair as possible, hence we spread uniformly, giving us probabilities $2/12 = 1/6$. For $p^*_{U \cup B}$, there are 2 unaccounted 1s in 6 entries, giving us $2/6 = 1/3$. Finally, all 1s are accounted for in $p^*_{T \cup U \cup B}$. Outside of $T \cup U \cup B$ we have no information, hence these probabilities default to $1/2$.

### 4 Comparing Sets of Tiles

Now that we have a technique for incorporating the information contained within a set of tiles into a model, we can use these models to compare sets of tiles.

We assume that we are given three tile sets $T$, $U$, and $B$. Let tile set $B$ contain the background information. Typically, this information would be simple, like column margins, row margins, or just the proportions of ones in the whole dataset. $B$ can be also be empty, if we do not have or wish to use any background knowledge. Our goal is now to compute the distance between $T$ and $U$ given $B$. We assume that the frequencies for the tile sets we are given are mutually consistent; which is automatically guaranteed if the frequencies for all three tile sets are computed from a single dataset. Now, let $M = T \cup U \cup B$ be the collection containing all tiles. We define the distance between $T$ and $U$, w.r.t. $B$, as

$$d(T, U; B) = KL(M \parallel U \cup B) + KL(M \parallel T \cup B) \over KL(M \parallel B).$$

Using Theorem 2 we can compute the distance in $O(NM)$ time.

If the given tiles are exact, the distance has a simple interpretable form; namely, the distance can be expressed with Jaccard similarity.

Theorem 5. Assume three tile collections $T$, $U$, and $B$ with exact frequencies $\{\alpha_T\}$, $\{\beta_U\}$, and $\{\gamma_B\}$. Define $X = s(T) \setminus s(B)$ and $Y = s(U) \setminus s(B)$. Then $d(T, U; B) = 1 - |X \cap Y|/|X \cup Y|$. 

Example 3. Let us continue Example 2. To compute $d(T, U; \emptyset)$ we first note that $T$ and $U$ only have exact tiles, and hence we can use Theorem 5. So, we have
\[|s(T) \setminus s(U)| = 2, \ |s(U) \setminus s(T)| = 8, \text{ and } |s(T) \cup s(U)| = 18.\] And hence, the distance \(d(T, U; \emptyset) = (2 + 8)/18 = 5/9.\)

Next, let \(M = T \cup U \cup B.\) To compute \(d(T, U; B),\) note that
\[KL(M \parallel T \cup B) = 2 \log(6) + 10 \log(6/5) \approx 5.4067.\]

where the first term represents the positive entries in \(M\) and the second term the negative entries in \(M.\) Similarly, \(KL(M \parallel B) \approx 15.2,\) and \(KL(M \parallel U \cup B) \approx 3.8.\) Consequently, the distance is equal to \(d(T, U; B) \approx 0.6\) which is slightly larger than \(d(T, U; \emptyset) \approx 0.56.\) This is due to the fact that adding \(B\) to \(T\) makes the probability of encountering a 1 at \((3, 4)\) and \((3, 5)\) in the model less likely. Hence, given that background knowledge, and regarding \(U,\) we are more surprised to find that these entries indeed contain ones.

5 Redescribing Sets of Tiles

Above, we were only concerned in finding out how much information two tile sets share. In this section we consider a more elaborate problem. Namely, given two tile sets, say \(T\) and \(U,\) we want to find out which tiles from \(U\) best describe the information provided by \(T.\) To this end, we will use our distance as follows.

\[\text{Problem 1 (Redescribe). Given three sets of tiles } T, U, \text{ and } B \text{ with consistent frequencies, find a subset } V \subseteq U \text{ such that } d(V; T; B) \text{ is minimized.}\]

It turns out that finding the best tile subset is computationally intractable.

\[\text{Theorem 6. The decision version of Redescribe is an NP-hard problem.}\]

Hence, we resort to a simple greedy heuristic: we add iteratively a tile that makes the current tile set closest to the target tile set. We stop the algorithm when we can no longer decrease the distance by adding more tiles.

6 Related Work

To our knowledge, defining a distance between two general tile sets is a novel idea. However, there exist several techniques for comparing datasets using patterns which comes to comparing the same pattern set with different supports. Such proposals include a Mahalanobis distance between itemset collections [23] and a compression-based distance between itemsets [25]. In addition, Hollmén et al. suggested using \(L_1\) distance between frequent itemset collections, where the missing frequencies were estimated with the support threshold [11].

From technical point of view, comparing pattern sets given background knowledge is akin to defining an interestingness measure based on deviation from the background knowledge. In fact, our approach for building a global Maximum Entropy model from tiles was inspired by the work of De Bie [5], where he builds a similar maximum entropy model from row and column margins (i.e. a
Rasch model [21]) and uses it as a static null hypothesis to rank tiles. Further related proposals include iterative mining of patterns by empirical p-values and randomisation [10], and maximum entropy models based on itemsets [12, 27].

Several techniques have been proposed for mining sets of tiles. Geerts et al. suggested discovering tilings that cover as many ones as possible [8]. Xiang et al. gave a method to mine (possibly noisy) tiles that cover ones while minimising a cost: the number of transactions and items needed to describe tiles [28]. These methods focus on covering the ones in the data, alternatively, we can assess the quality of a tiling by statistical means. Gionis et al. suggested discovering hierarchical tiles by building a statistical model and optimising an MDL score [9]. De Bie gave a maximum entropy model based on column/row margins to rank tiles [5]. Vreeken et al. propose that the best set of tiles (or itemsets) is the tile set that compresses the dataset best [26].

An alternative approach for discovering tiles is to consider a Boolean matrix factorisation [14]. That is, factorise the dataset into two low rank Boolean matrices, where the row vectors of the one matrix correspond to itemsets, while the column vectors of the other matrix correspond to tid-lists. The Boolean product of these matrices naturally defines a set of noisy tiles.

Compared to tiles, computing maximum entropy models based on itemsets is much more difficult. The reason for this is that there is no equivalent version of Lemma 1 for itemsets. In fact, computing an expected value of an itemset from a maximum entropy model is \texttt{PP}-hard [22]. To avoid these problems, we can convert itemsets to exact tiles by considering their supporting transactions.

Redescribing tile sets is closely related to redescription mining, in which the idea is to find pairs of syntactically different patterns covering roughly the same transactions. Ramakrishnan et al. [20] originally approached the problem by building decision trees. Other approaches include Boolean Formulae with no overlap [7], and exact minimal redescriptions [29]. From a computational point of view, the difference between our problem and existing work is the goal: redescription mining aims to construct an alternative pattern given a single target pattern, while we consider sets of target and candidate patterns, and aim to find the subset of patterns from candidates that together describe the target best.

7 Experiments

In this section we empirically evaluate our measure. We provide our code for research purposes\(^1\). We evaluate our measure on four publicly available real world datasets. Abstracts contains the abstracts of the papers accepted at ICDM up to 2007, where words have been stemmed and stop words removed [5]. The DNA amplification data contains information on DNA copy number amplifications. Such copies are known to activate oncogenes and are the hallmarks of nearly all advanced tumours [17]. The Mammals presence data consists of presence records of European mammals\(^2\) within geographical areas of 50 × 50 kilome-

\(^1\) http://www.adrem.ua.ac.be/implementations/
\(^2\) Available for research purposes: http://www.european-mammals.org
Table 1. Number of tiles extracted by each of the considered methods.

| Dataset     | N  | M   | clust | bicl | atcl | sscl | asso | tiling | hyper | itt | krimp | mtv |
|-------------|----|-----|-------|------|------|------|------|--------|-------|-----|-------|-----|
| Abstracts   | 859| 3933| 5 × M | 25   | 753  | 100  | 100  | 38     | 100   | 100 | 25    |
| DNA         | 4590| 392 | 5 × M | 25   | 56   | 100  | 100  | 32     | 100   | 100 | 100   |
| Mammals     | 2183| 124 | 5 × M | 25   | 28   | 100  | 91   | 3      | 100   | 2   | 100   | 14  |
| Paleo       | 501 | 139 | 5 × M | 25   | 514  | 100  | 100  | 100    | 100   | 71  | 85    | 14  |

7.1 Methods and Mining Results

We apply our measure to compare between the results of ten different exploratory data mining methods for binary data. Table 1 gives an overview, here we state the parameters we use and how we refer to each of the methods between brackets.

We employ simple $k$-means clustering (clus) with $k = 5$ clusters, using $L_1$ distance. We turn the clusters into tiles by computing column margins inside each cluster. A bi-clustering simultaneously cluster the transactions and the items; a cluster is then a tile defined by the corresponding pair of item and transaction clusters [18]. We apply biclustering (bicl) by separately clustering the columns and rows using again $k$-means clustering ($k = 5$) and combine the two clusterings into a grid. Puolamäki et al. showed that a good approximation bound can be achieved with this approach [19]. Each cluster is represented by a single tile. We use the parameter-free attribute clustering (atcl) approach by Mampaey & Vreeken [13], and convert each cluster into a tile — thereby somewhat oversimplifying these results. For subspace clustering (sscl), we used the implementation of Müller et al. [16] of the ProClus algorithm [1], mined 100 clusters, each over maximally 32 dimensions, and converted each into a noisy tile. We mined overlapping Tilings [8] of up to 100 exact tiles (tiling), allowing the algorithm a maximum of 8 hours. The Asso algorithm [14], (asso), was ran with a maximum of 100 factors, of which the non-empty ones were converted into tiles.

Per dataset, we mined up to 100 hyper rectangles [28], (hyper), as noisy tiles. We mined Information-Theoretic exact Tiles [5], (itt), where the method automatically selects the number of tiles, however, we used top-100 tiles, at most. For mining Maximum-Entropy Tiles (mtv) [12], we set a maximum of 2 hours and 100 tiles. We used Krimp [26] to mine itemsets that compress, and took the top-100 most-used itemsets as tiles using their Krimp-usage as tid-sets (krimp). All four of these methods select itemsets from a candidate collection, for which we used closed frequent itemsets mined at as low as feasible support thresholds, of resp. 5, 1, 800, and 1. For Krimp, however, we could use lower support thresholds for the Abstract and Mammals datasets, resp. 4 and 300.

3 NOW public release 030717 available from [6].
7.2 Measuring Distances

First, we evaluate whether the measured distance converges to 0 when two sets of tiles approximate each other. To this end, we take the tile sets of asso, krimp, and itt, as obtained on resp. the Abstracts and DNA datasets. In Figure 2 we plot, per method, the measured distance between the top-$k$ and top-100 tiles. We give the measurements for three different background knowledge settings, resp. no background knowledge, knowledge of the average density of the dataset, and the column margins. The tiles are sorted according to their output order, for asso and itt, and ascending on code length for krimp.

As Figure 2 shows, measurements indeed converge to 0 for higher $k$, i.e. when the two tile sets become more identical. Adding background information typically increases the distance. This is due to two reasons. First, when density is used, then we can infer additional differences between the areas that are not covered by tiles, thus highlighting the differences. Second, when we are using column margins, we reduce $KL(M \parallel B)$, the joint information w.r.t. the background knowledge, consequently increasing the distance. Interestingly enough, for Abstracts the distances for asso decrease when density is used. This is caused by the fact that asso produces many overlapping tiles and these overlaps are emphasised when density is used.

7.3 Distances between Results

Our main experiment is to investigate how well we can compare between results of different methods. To do so, for every dataset, and every method considered, we convert their results into tile sets as described above. We measure the pairwise difference between each of these tile sets, using resp. the empty set, overall density, the column margins, and the combination of row and column margins, as background knowledge. For analysis, we present these numerical results, and the
### Fig. 3. Sammon projections of distances between tile sets. Each row represents a dataset and each column represents used background knowledge. Note that `atcl` and `mtv` are not included in the rightmost columns, as their tiles provide no information beyond column margins.

```plaintext
- **Palaeo**
  - **Average**: \( B = \emptyset \)
  - **Density**: \( B = \text{density} \)
  - **Columns**: \( B = \text{columns} \)
  - **Columns, Rows**: \( B = \text{columns, rows} \)

|   | hyper | bicl | krimp | sscl | asso |
|---|-------|------|-------|------|------|
| Paleo |   +   |   +  |   +   |   +  |   +  |
| Abstracts |   +   |   +  |   +   |   +  |   +  |
| DNA |   +   |   +  |   +   |   +  |   +  |
| Mammals |   +   |   +  |   +   |   +  |   +  |
| Average |   +   |   +  |   +   |   +  |   +  |
```

**average**

\( B = \emptyset \)

\( B = \text{density} \)

\( B = \text{columns} \)

\( B = \text{columns, rows} \)

**averages over the datasets**, visually in Figure 3 by plotting all pairwise distances by Sammon projection. We colour tiling and clustering methods differently.

Considering the first column first, we see that without background knowledge three of the clustering approaches provide virtually the same result. We also see that, albeit not identical, the results of `asso`, `itt`, `krimp`, and `tiling` are relatively close to each other; which makes sense from a conceptual point of view, as these methods are methodologically relatively similar. For DNA, the measured dissimilarity between these methods lies between 0.28 and 0.38, whereas the dissimilarities to the other methods measure approximately 0.9.

We observe that `hyper`, while conceptually similar, is measured to provide different results when no background knowledge is given. This is mostly due to it including a few very large tiles, that practically cover the whole data, whereas
7.4 Redescribing Results

Next, we empirically evaluate how our measure can be employed with regard to redescribing results; both as validation as well as possible application.

To this end, we first investigate the redescription of results of completely different methods. As such, we take clustering as the target and density as the background information, and redescribe its result by using the tile sets of five of the pattern mining methods as candidate tile sets. Table 2 shows the results of these experiments: for four datasets, the measured divergence between the redescription and the target, the divergence of the complete tile set to the target, and the number of tiles selected for the redescription. First, and foremost, we see that by redescription the measured divergence decreases, which correctly shows that by filtering out, e.g. too specific, tiles that provide information not in the target, we obtain a better description of the target.

We also see, with the exception of Mammals, that the measurements are quite high overall, suggesting the clustering provides information these results

| Dataset | Redescription / Full Tile Set (# of Tiles) |
|---------|-----------------------------------|
| Abstracts | 0.76/0.76 (70) | 0.74/0.74 (38) | 0.50/0.57 (32) | 0.68/0.68 (100) | 0.85/0.85 (98) |
| DNA | 0.65/0.83 (11) | 0.70/0.79 (9) | 0.67/0.84 (21) | 0.67/0.81 (11) | 0.67/0.81 (19) |
| Mammals | 0.30/0.31 (51) | 0.68/0.68 (3) | 0.34/0.39 (24) | 0.78/0.78 (2) | 0.49/0.49 (91) |
| Paleo | 0.68/0.83 (16) | 0.69/0.78 (28) | 0.68/0.81 (23) | 0.73/0.81 (21) | 0.74/0.79 (38) |

**Table 2.** Redescribing the results of clustering (clus). Measurements for the found redescription, and complete tile set, and the number of selected tiles.
do not; not surprising, as these pattern mining methods focus on covering 1s, and not necessarily cover the whole data. Indeed, we see that by providing large and noisy tiles, and so covering more of the data, *asso* and *hyper* lead to the best redescriptions of the clustering results. In particular for *Mammals*, the target can be approximated very well, by resp. only half and a quarter of the total tiles. Overall, we note that typically only fractions of the full tile sets are selected, yet the amount of shared information is larger than for the full tile set: the pattern mining methods provide detailed local information not captured by clustering.

Second, we take a closer look at individual redescriptions. In order to be able to interpret these, we use the *Abstracts* dataset. We use *asso*, *krimp*, and *itt*, as these provide sufficiently many tiles to choose from; we leave *hyper* out, as for this data it mostly gives only very general tiles, covering all 1s in only 100 tiles.

By hand, we select 5 out of 100 *krimp* tiles, and we identify, for *asso* and *itt*, the sets of tiles that best approximate that partial result, and investigate how well the target concepts are approximated. In Figure 4, we give an example. By the high distances, 0.77 and 0.83, we see the target is not approximated in detail. Overall, for *itt* we find only high-level translations that leave out detail, as its full tile set consists mostly of small itemsets. For *asso*, we see the redescription consists of target tiles combined with general concepts, which together give a reasonable approximation of the target. It is important to note that not simply all intersecting itemsets are given, but only those that provide sufficient information on the target; for both methods, overly large and overly general tiles (e.g. ‘high’) are not included in the redescription.

8 Discussion

The experimental evaluation of our measure shows it works well in practice, providing insightful groupings of (the results of) exploratory data mining methods on binary data, and meaningful redescriptions of partial results.

The goal of this paper is to take a first step towards comparing between the results of different data mining methods. The method we propose here is for results obtained on binary data. By developing further maximum entropy modelling techniques, however, the same basic idea could be applied to richer data types. We currently straightforwardly convert results into tile sets, capturing much of the information they provide. Ideally, however, we would be able to encode structure beyond simple tile densities, e.g. which attribute-value combinations occur how often (i.e. for *atcl*), such that the information captured in a data mining result can be maintained even more precisely when converted into sets of tiles. Such more complex modelling aside, the general idea of comparing how many possible datasets exhibit such structure remains the same.

Besides measuring divergence of results between different methods, our approach can also be used to choose the most informative result out of many of one randomised method, or, to measure differences between results when varying parameters of a method. It is important to note that our method solely measures the information shared between two sets of tiles; it does not measure the
subjective quality of results, nor does it say anything about the ease of analysis of a result. Instead, it gives insight whether or not a, possibly easily interpreted, result is as informative as another, possibly much more complex result.

9 Conclusion

In this paper we discussed comparing results of different explorative data mining algorithms. We argued that any mining result identifies some properties of the data, and that, in an abstract way, we can identify all datasets for which these properties hold. By incorporating these properties into a model using the Maximum Entropy principle, we can measure the shared amount of information by Kullback-Leibler divergence. The measure we construct this way is flexible, and naturally allows including background knowledge, such that differences in results can be measured from the perspective of what a user already knows.

As a first step towards comparing results in general, we formalised our approach for binary data, showed results are easily converted into tiles, and discussed how to incorporate these into a Maximum Entropy model. Our approach provides a means to study and tell differences between results of different data mining methods. As an application, we showed it can be used to parameter-free identify which parts of results best redescribe a given result. Experiments showed our measure gives meaningful results, correctly identifies methods that are similar in nature, and automatically identifies sound redescriptions of results.

Acknowledgements

The authors wish to thank, in alphabetical order: Tijl De Bie for his implementation of [5]; David Fuhry for his implementation of [28]; Stijn Ligot for running experiments with ProClus [1, 16]; Michael Mampaey for the implementations of [13] and [12]; Pauli Miettinen for his implementaiton of Asso [14].

Nikolaj Tatti and Jilles Vreeken are supported by Post-Doctoral Fellowships of the Research Foundation – Flanders (FWO).

References

1. C. C. Aggarwal, J. L. Wolf, P. S. Yu, C. Procopiuc, and J. S. Park. Fast algorithms for projected clustering. In Proc. ACM SIGMOD’99, pages 61–72. ACM, 1999.
2. T. M. Cover and J. A. Thomas. Elements of Information Theory, 2nd ed. 2006.
3. I. Csiszár. I-divergence geometry of probability distributions and minimization problems. Ann. Prob., 3(1):146–158, Feb. 1975.
4. J. Darroch and D. Ratcliff. Generalized iterative scaling for log-linear models. Ann. Math. Stat., 43(5):1470–1480, 1972.
5. T. De Bie. Maximum entropy models and subjective interestingness: an application to tiles in binary databases. Data Min. Knowl. Disc., 2010.
6. M. Fortelius, A. Gionis, J. Jernvall, and H. Mannila. Spectral ordering and biochronology of european fossil mammals. Paleobiology, 32(2):206–214, 2006.
7. A. Gallo, P. Miettinen, and H. Mannila. Finding subgroups having several descriptions: Algorithms for redescription mining. In Proc. SDM’08, 2008.
8. F. Geerts, B. Goethals, and T. Mieliikkäinen. Tiling databases. In Proc. DS’04, pages 278–289, 2004.
9. Aristides Gionis, Heikki Mannila, and Jouni K. Seppänen. Geometric and combinatorial tiles in 0-1 data. In Proc. ECML PKDD’04, pages 173–184, 2004.
10. S. Hanhijärvi, M. Ojala, N. Vuokko, K. Puolamäki, N. Tatti, and H. Mannila. Tell me something I don’t know: randomization strategies for iterative data mining. In Proc. KDD’09, pages 379–388, 2009.
11. J. Hollmén, J. K. Seppänen, and H. Mannila. Mixture models and frequent sets: combining global and local methods for 0-1 data. In Proc. SDM’03, 2003.
12. M. Mampaey, N. Tatti, and J. Vreeken. Tell me what I need to know: succinctly summarizing data with itemsets. In Proc. KDD’11, 2011.
13. M. Mampaey and J. Vreeken. Summarising data by clustering items. In Proc. ECML PKDD’10, pages 321–336, 2010.
14. P. Miettinen, T. Mieliikkäinen, A. Gionis, G. Das, and H. Mannila. The discrete basis problem. IEEE Trans. Knowl. Data Eng., 20(10):1348–1362, 2008.
15. A.J. Mitchell-Jones, G. Amori, W. Bogdanowicz, B. Krystufek, P.J. H. Reijnders, F. Spitzberger, M. Stubbe, J.B.M. Thissen, V. Vohralík, and J. Zima. The Atlas of European Mammals. Academic Press, 1999.
16. E. Müller, S. Günneweg, I. Assent, and T. Seidl. Evaluating clustering in subspace projections of high dimensional data. In Proc. VLDB’09, 2009.
17. S. Myllykangas, J. Himberg, T. Böling, B. Nagy, J. Hollmén, and S. Knuutila. DNA copy number amplification profiling of human neoplasms. Oncogene, 25(55):7324–7332, 2006.
18. R. Pensa, C. Robardet, and J-F. Boulicaut. A bi-clustering framework for categorical data. In Proc. ECML PKDD’05, pages 643–650, 2005.
19. K. Puolamäki, S. Hanhijärvi, and G. C. Garriga. An approximation ratio for biclustering. Inf. Process. Lett., 108(2):45–49, 2008.
20. N. Ramakrishnan, D. Kumar, B. Mishra, M. Potts, and R. F. Helm. Turning cartwheels: an alternating algorithm for mining redescriptions. In Proc. KDD’04, pages 266–275, 2004.
21. G. Rasch. Probabilistic Models for Some Intelligence and Attainment Tests. Danmarks paedagogiske Institut, 1960.
22. N. Tatti. Computational complexity of queries based on itemsets. Inf. Process. Lett., pages 183–187, 2006.
23. N. Tatti. Distances between data sets based on summary statistics. J. Mach. Learn. Res., 8:131–154, 2007.
24. N. Tatti and J. Vreeken. Comparing apples and oranges - measuring differences between exploratory data mining results. Technical Report 2011/03, University of Antwerp, 2011.
25. J. Vreeken, M. van Leeuwen, and A. Siebes. Characterising the difference. In Proc. KDD’07, pages 765–774, 2007.
26. J. Vreeken, M. van Leeuwen, and A. Siebes. Krimp: Mining itemsets that compress. Data Min. Knowl. Disc., 23(1):169–214, 2010.
27. C. Wang and S. Parthasarathy. Summarizing itemset patterns using probabilistic models. In Proc. KDD’06, pages 730–735, 2006.
28. Y. Xiang, R. Jin, D. Fulry, and F. Dragan. Summarizing transactional databases with overlapped hyperrectangles. Data Min. Knowl. Disc., 2010.
29. M.J. Zaki and N. Ramakrishnan. Reasoning about sets using redescription mining. In Proc. KDD’05, pages 364–373, 2005.
A Proofs for Theorems

Proof (of Theorem 2). Let $p^*$ be the maximum entropy distribution. Define a distribution $q$

$$q(D) = \prod_{i,j} p^*((i,j) = D(i,j)).$$

Lemma 1 now implies that $q \in \mathcal{P}$. Assume that $p^*(D) > 0$. Then $p^*((i,j) = D(i,j)) > 0$ and consequently $q(D) > 0$. This implies that $\{D \mid p^*(D) = 0\} = \{D \mid q(D) = 0\}$.

Fix $(i,j)$ such that $0 < p^*((i,j) = 1) < 1$. Let $D$ be a dataset and let $D'$ be the dataset with flipped $(i,j)$th entry. Then $q(D) > 0$ if and only if $q(D') > 0$ and hence $p^*(D) > 0$ if and only if $p^*(D') > 0$. Divide the dataset space

$$D_v = \{D \in D \mid D(i,j) = v, p^*(D) > 0\}.$$

The previous argument shows that for each $D \in D_1$, the dataset $D'$ with flipped entry $D' \in D_0$. Theorem 1 implies that $p^*(D)/p^*(D') = \exp(\sum_{T \in \mathcal{T}((i,j))} \lambda_T)$.

$$p^*((i,j) = 1)/p^*((i,j) = 0) = \sum_{D \in D_1} p^*(D)/\sum_{D \in D_0} p^*(D) = \sum_{D \in D_1} p^*(D)/\sum_{D \in D_1} p^*(D') = \exp(\sum_{T \in \mathcal{T}((i,j))} \lambda_T).$$

The theorem follows by rearranging the terms.

Proof (of Theorem 3). This proof follows the same lines as the proof in [4] with some additional handling of zero probabilities. To prove the theorem we need the following technical lemma.

Lemma 2. Let $\{p^k\}$ be a sequence of distributions over a finite domain converging to a limit distributions $p$. Assume that there is a set $X$ such that for $x \in X$ each $p^k$ has the form

$$p^k(x) = \exp(r_{0k} + \sum_{i=1}^N r_{ik} S_i(x)),$$

for a set of functions $\{S_i\}$ and weights $r_{ik}$. Assume further that $p^k(x) = 0$ if $x \notin X$. Let $h_i = \mathbb{E}_p[S_i]$. Let $q$ be a distribution such that $\mathbb{E}_q[S_i] = h_i$ for $i = 1, \ldots, N$, and $q(x) = 0$ for $x \notin X$. Then $H(p) \geq H(q)$.

Proof. For simplicity, write $S_0(x) = 1$. Consider series

$$a_k = \sum_{x \in X} p(x) \log p^k(x) = \sum_{x \in X} p(x) \sum_{i=0}^N r_{ik} S_i(x) = \sum_{i=0}^N r_{ik} h_i$$

$$= \sum_{x \in X} q(x) \sum_{i=0}^N r_{ik} S_i(x) = \sum_{x \in X} q(x) \log p^k(x).$$
Taking the series \( \{a_k\} \) to the limit leads us to
\[
H(p) = -\lim_k a_k = -\lim_k \sum_{x \in X} q(x) \log p^k(x) = -\sum_{x \in X} q(x) \log p(x)
\].

The last equality holds because each term of the sum is negative and the divergence of a single term would imply that \( H(p) = \infty \). Hence, either \( q(x) = 0 \) or \( \lim_k p^k(x) > 0 \) which guarantees the convergence. Now
\[
0 \leq KL(q \parallel p) = -H(q) - \sum_{x} q(x) \log p(x) = -H(q) + H(p)
\]
which proves the lemma.

Let \( q \in \mathcal{P} \) be a distribution satisfying the frequencies.

Consider the first phase of the algorithm. Let \( T \) be an exact tile and let \((i,j) \in s(T)\). Then \( q((i,j)=1) = \alpha_T \), otherwise \( fr(T; q) \neq \alpha_T \). The first phase sets \( p((i,j)=1) = \alpha_T \). Moreover, \( p((i,j)) \) will remain unchanged during the second phase since the update equation does not change \( p((i,j)=1) \) if it is equal to 0, 1.

Consider the second phase of the algorithm. Write \( T' \) to be the subset of \( T \) containing all noisy tiles. Let \( p^k \) be the distribution right before the \( k \)th iteration and similarly let \( \alpha_T^k \) to be the frequency of \( T \) right before the update of tile \( T \) during the \( k \)th iteration.

We claim that \( 0 < \alpha_T^k < 1 \). To see this, we note that the only way \( \alpha_T^k = 0,1 \) is that each entry of \( s(T) \) in \( p \) are simultaneously 0 or simultaneously 1. In other words, these entries have been set by the first phase. This again implies that \( \alpha_T = 0,1 \). Otherwise, target frequencies are inconsistent. But this means that \( T \) is not processed during the second phase which is a contradiction. Let us abuse notation and write
\[
KL(x \parallel y) = x \log \frac{x}{y} + (1-x) \log \frac{1-x}{1-y},
\]
where \( 0 \leq x, y \leq 1 \). A straightforward calculation reveals that the updates during the \( k \)th iteration are equivalent to
\[
p^{k+1}(D) = p^k(D) \prod_{T \in T'} \left( \frac{\alpha_T}{\alpha_T^k} \right)^{fr(T; D)} \left( \frac{1 - \alpha_T}{1 - \alpha_T^k} \right)^{1-fr(T; D)}.
\]

Then
\[
KL(q \parallel p^{k+1}) = KL(q \parallel p^k) - \sum_{T \in T'} \alpha_T \log \frac{\alpha_T}{\alpha_T^k} + (1 - \alpha_T) \log \frac{1 - \alpha_T}{1 - \alpha_T^k}
\]
\[= KL(q \parallel p^k) - \sum_{T \in T'} KL(\alpha_T \parallel \alpha_T^k).
\]
Hence, the series \( KL(q \parallel p^{k+1}) \) are decreasing and, since they are bounded from below, converging. This implies that \( KL(\alpha_T \parallel \alpha_T^k) \to 0 \) which in turn implies that \( \alpha_T^k \to \alpha_T \).
Since \( p^k \) is a sequence of distributions in a compact space, there is a subsequence converging to a limit, say \( p' \). By setting \( X = \{ D \mid p'(D) > 0 \} \) we can use Lemma 2 to show that \( p' \) has the highest entropy among the distributions in \( \mathcal{P} \). Since the maximum entropy distribution is unique all subsequences converge to \( p^* \). □

**Proof (of Theorem 4).** We will prove the theorem by showing that \( p^* \) satisfies the conditions in Theorem 2. Let \( \mathcal{P} \) be the distributions satisfying the frequencies. Let \( \mathcal{Z} \) be the collection of datasets for which \( p(D) = 0 \) for every \( p \in \mathcal{P} \). Set \( \mathcal{Z}' = \{ D \mid p^* (D) = 0 \} \). Let \( q \in \mathcal{P} \) and let \( (i, j) \in s(T) \). Corollary 1 implies that \( q((i, j) = 1) = \alpha_T \). This implies that \( \mathcal{Z}' \subseteq \mathcal{Z} \) and since \( p^* \in \mathcal{P} \), we must have \( \mathcal{Z} = \mathcal{Z}' \). By setting \( \lambda_T = 0 \) for every tile we see that \( p^* \) has the exponential form given in Theorem 2. Hence, \( p^* \) is truly the maximum entropy distribution. □

**Theorem 5** follows directly from the following lemma.

**Lemma 3.** Let \( T \) and \( U \) be two tile sets such that \( s(U) \subseteq s(T) \). Then \( KL(T \parallel U) = |s(T) \setminus s(U)| \log 2 \).

**Proof.** Let \( X = s(T) \setminus s(U) \). Theorem 4 implies that

\[
KL(T \parallel U) = \sum_{i, j} \sum_{v=0,1} p^*_T((i, j) = v) \log \frac{p^*_T((i, j) = v)}{p^*_U((i, j) = v)}
\]

The only non-zero terms in the sum are the entries \( (i, j) \in X \) for which \( p^*_T((i, j) = v) = 1 \). Hence we have

\[
KL(T \parallel U) = \sum_{(i, j) \in X} 1 \times \log \frac{1}{1/2} = |X| \log 2
\]

This proves the lemma. □

**Proof (of Theorem 6).** We prove the hardness by reducing \textsc{ExactCover}, testing whether a cover has a mutually disjoint cover, to the decision version of \textsc{Redescribe}.

Assume that we are given a ground set \( S \) with \( N \) elements and a cover \( \mathcal{C} = \{ C_1, \ldots, C_L \} \) for \( S \). To define the tiles and the target frequencies we will first define a particular dataset \( D \). This dataset has \( N + 1 \) transactions of length \( N \), an item representing an item in the ground set \( S \). The first \( L \) transactions correspond to the cover sets, \( (i, j) \) entry is equal to 1 if and only if \( C_i \) contains the \( j \)th item. The last \( N + 1 \) transactions are full of ones.

The tile sets are as follows. \( B = \emptyset \). \( T \) contains one tile with the last \( N + 1 \) transactions and all items. \( U = \{ U_1, \ldots, U_L \} \) contains \( L \) tiles such that \( a(U_i) = C_i \) and \( t(U_i) = \{ i, L + 1, \ldots, L + N + 1 \} \). Let \( \mathcal{V} \) be a subset of \( \mathcal{U} \). Write \( f(\mathcal{V}) = \sum_{V \in \mathcal{V}} |a(V)| \). Theorem 5 now implies that

\[
d(\mathcal{V}, T; B) = \frac{(N + 1)(N - |a(\mathcal{V})|) + f(\mathcal{V})}{(N + 1)N + f(\mathcal{V})}.
\]
If $|a(\mathcal{V})| < N$, then we can augment with $V$ with a tile containing an item outside of $a(\mathcal{V})$. This will reduce the first term in the numerator at least by $N+1$ and increase the second term by $N$, at maximum. Thus such augmentation will decrease the distance and so we can safely assume that $|a(\mathcal{V})| = N$. This implies that $d(\mathcal{V}, T; B) = \frac{f(\mathcal{V})}{(N+1)N + f(\mathcal{V})}$. This is a monotonic function of $f(\mathcal{V})$. Since $a(\mathcal{V})$ contains all items, $f(\mathcal{V}) \geq N$ and is equal to $N$ if and only if the tiles in $\mathcal{V}$ are disjoint. Hence ExactCover has a solution if and only if, there is a redescription with distance equal to $N/(N(N+1) + N)$. □