Adaptive Multi-model Approaches to Pattern Set Mining

Tatiana Makhalova
University of Lorraine,
CNRS, Inria, LORIA,
F-54000 Nancy, France
tatiana.makhalova@inria.fr

Sergei O.Kuznetsov
National Research University
Higher School of Economics,
Moscow, Russia
skuznetsov@hse.ru

Amedeo Napoli
University of Lorraine,
CNRS, Inria, LORIA,
F-54000 Nancy, France
amedeo.napoli@loria.fr

Abstract

Pattern Mining (PM) has an important place in Data Mining and Knowledge Discovery and has many applications in a wide variety of domains. To date, a lot of different approaches to PM have been proposed. However, new methods continue to appear. Some of the reasons for that are the following: (i) there is no gold standard for evaluating the quality of PM approaches, (ii) the results of existing approaches are unsatisfactory. But what is wrong with them? In this paper, we adopt the best practices of building supervised models to PM. We propose a PM method KeepItSimple that combines the technique of supervised learning and the state-of-the-art of modern PM. We show in experiments that the proposed approach allows for obtaining small sets of interesting and non-redundant patterns.

1 Introduction

A generic objective of PM is to discover a small set of non-redundant and interesting patterns that describe together a large portion of data and that can be easily interpreted [1]. Early approaches to PM, called *static* [18], were based on unchangeable (static) criteria, i.e., interestingness measures. For example, in frequent pattern mining, interesting patterns are those whose frequency exceeds a particular threshold. The main drawbacks are that instead of computing an "interesting" set of most interesting patterns, these approaches return a set of independently interesting patterns, which are often very similar (lack of diversity). Moreover, the choice of a measure is quite subjective and most of the time it is not easy to provide an explanation or a justification about using one measure rather than some others [9].

In "modern approaches" interestingness is assessed by informativeness [6], where the interestingness measure changes during mining, explaining why these approaches are called *dynamic* [18]. Instead of computing independently interesting patterns, they compute an "interesting set of patterns", meaning that each pattern should be interesting not only w.r.t. a dataset, but also w.r.t. other patterns in the set. These approaches are based on either the principle of maximum entropy [2, 7, 12] or the Minimum Description Length principle (MDL) [5, 16, 17]. The majority of "modern approaches" to PM is based on MDL. They are usually performed in two steps: (i) computing candidate to optimal patterns (i.e., reducing the exponential-size pattern search space), (ii) selecting greedily those patterns that minimize the total description length.

In: Rudolph S., Marreiros G. (eds.): Proceedings of STAIRS, Santiago de Compostela, Spain, august 2020, published at http://ceur-ws.org

All the existing MDL-based approaches compute only one model, i.e., search for a set of mutually optimal (nonredundant) patterns. In [16], based on the assumption that a dataset may contain several models, the authors propose to compute a sequence of MDL-optimal pattern sets at different levels of specification. They introduce a structural function over datasets. The structural function maps a natural number $k \in \mathbb{N}$ into a set of k MDL-optimal patterns. All pattern sets are computed independently of each other, and the model (pattern set) of size k may differ by more than one pattern from the model of size $k+1$. However, the patterns from different models (pattern sets) may be similar (or repetitive), thus, the whole set of models may be redundant.

The idea of using several models to describe data is widely used in supervised methods. The closest to PM class of supervised learning methods is the one based on decision trees [15]. A label of a tree node can be interpreted as an element of a pattern, thus a pattern is a set of labels in the path from the root to a leaf. All modern tree-based approaches are based on ensembles of trees, e.g., random forests [3], extremely randomized trees [10], boosted trees [8], etc.

The tree-based methods are known to have the following properties: (i) an ensemble of trees is better than a single decision tree, (ii) an ensemble of "shallow" trees outperforms an ensemble of deeper trees, (iii) an ensemble of boosted trees (an ensemble of trees where each new tree is focused on the examples misclassified by the previous trees) outperforms other models of tree ensembles. These rules of thumb work well in general, for a wide variety of datasets.

PM approaches have never been considered from this perspective. Based on the assumption that the best practices of supervised learning may work well in unsupervised settings, we propose to adapt the aforementioned principles for building an ensemble of classifiers to PM approaches. As a basic method for computing a pattern set, we consider the state-of-the-art MDL-based PM approach called Krimp [19]. This is an analogue of a decision tree model, meaning that both of them compute pattern sets (in case of decision trees a pattern corresponds to a path from the root to a leaf), where a pattern is computed depending on other patterns.

The proposed MDL-based multi-model approach (i) uses at most a quadratic number of "easily-derivable" candidates to optimal patterns, (ii) builds a sequence of models (pattern sets), where each successive model describes the data fragments that have been insufficiently well described by previous models, (iii) evaluates interestingness of patterns by their informativeness based on MDL. Non-redundancy of patterns within a single model (in a pattern set) is ensured by MDL, while non-redundancy of patterns among models is ensured by a proposed data projection strategy, i.e., using only poorly described data fragments for computing the next model.

The paper has the following structure. In Section 2, we introduce the basic notations. In Section 3.1 we consider principles of computing simply-derivable itemsets (an analogue of shallow trees). In Section 3.2 we present an approach for computing a sequence of models (pattern sets), where each new model (pattern set) describes those regions of a dataset that are insufficiently well described by the previous models. In Section 4 we study the quality of the proposed approach in experiments. In Section 5 we conclude and give the direction of future work.

2 Basic Notions

2.1 Basic Notions of Pattern Mining

We work with transactional datasets and represent them as binary tables. In binary datasets, attributes are also called items, and patterns are called itemsets. As patterns we consider itemsets consisting of at least 2 attributes and describing at least 2 objects.

Let $I = \{i_1, \ldots, i_M\}$ be a set of attributes. A dataset $\mathcal{D}$ is a bag of N transactions over attributes in I, where each transaction t is a subset of I, i.e., $t \subseteq I$. The pattern search space consists of all possible subsets of I, i.e., it is the powerset $\mathcal{P}(I)$ of size $2^{|I|}$. An itemset X occurs in a transaction t, iff $X \subseteq t$, we call the set of objects where it occurs the *extent* of X, i.e., $ext(X) = \{t \in \mathcal{D} \mid X \subseteq t\}$. The frequency of X is the (relative) size of its image, i.e., $freq(X) = |ext(X)|$ (or $|ext(X)|/N$). An equivalence class $Equiv(X)$ of a pattern X is a set of itemsets with the same extent, $Equiv(X) = \{Y \in \mathcal{P}(I) \mid ext(Y) = ext(X)\}$. In our study we use *closed itemsets* as patterns. A *closed itemset* C is the maximal set of items common to a set of objects [13, 14], it is unique in its equivalence class. We call the closed itemset $C \in Equiv(X)$ the *closure* of X and denote it by $cl(X)$. Closed itemsets are a very common choice for candidates to MDL-optimal patterns, since (i) a closed itemset is the maximal set that represents all the itemsets with the same image, (ii) a closed itemset provides a lossless representation of these itemsets [14]. An example of a transactional dataset and the corresponding set of closed itemsets (ordered by the inclusion of attributes) is given in Figure 1, (a, b).

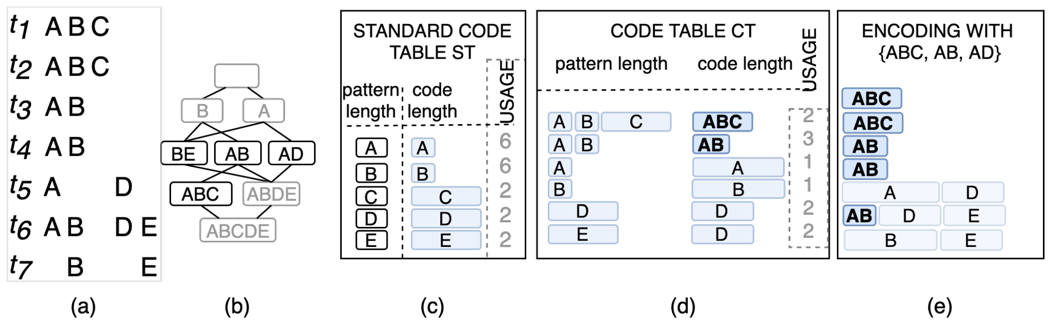


Figure 1: (a) A dataset, (b) partially ordered by inclusion of closed itemsets, patterns (with size and frequency of at least 2) are highlighted in bold, (c) the standard code table ST, (d) a code table containing patterns ABC and AB, (e) covering (encoding) with patterns ABC and AB.

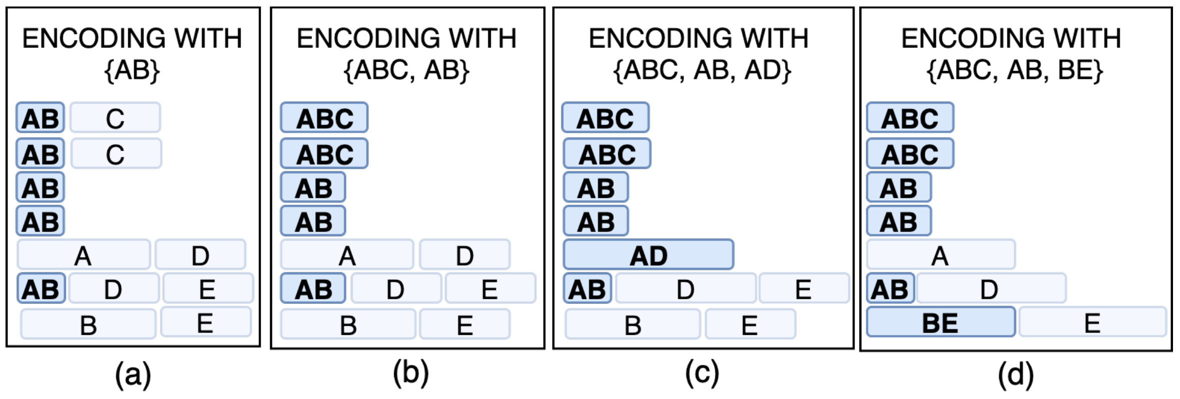


Figure 2: (a) adding AB (optimal), (b) adding ABC (optimal), (c) adding AD (not optimal), (d) adding BE (not optimal). The final set of optimal patterns is $\{ABC, AB\}$.

For the sake of simplicity, we denote itemsets by sequences of letters, e.g., ABC will be used to denote the attribute set $\{A, B, C\}$.

2.2 Minimum Description Length Principle in Pattern Mining

For a given dataset $\mathcal{D}$ and a set of models $\mathcal{M}$, the best model $M \in \mathcal{M}$ is the one that minimizes the total description length $L(\mathcal{D}, M) = L(M) + L(\mathcal{D}|M)$, where $L(M)$ is the length, in bits, of the description of M, and $L(\mathcal{D}|M)$ is the length, in bits, of the description of the data when encoded with M [11]. We consider how MDL is used in PM using the state-of-the-art method called Krimp [19]. In Krimp, the model M is represented by a two-column code table CT, where the left- and right-hand columns contain itemsets over attributes M and their associated prefix codes, respectively. The initial code table is called *standard code table* and denoted by ST. It contains only singletons (an example is given in Figure 1, c, and explained further). The code length of a singleton is inversely proportional to its frequency. By convention, the singletons are not patterns. A code table consists of two columns. The left-hand column contains patterns described in the attribute-wise manner, using the optimal codes from ST, i.e., the description length of a pattern X is given by $\sum_{X \in CT} L(X|ST)$. The length of an optimal code of pattern X is computed based on Shannon optimal codes, i.e., $L(code(X)) = -\log_2(P(X))$, where $P(X)$ is the probability of pattern X.

The probability of a pattern is based on its usage in a data covering (i.e., how many times a pattern is used to cover dataset). The probability of $X \in CT$ is given by

$$P(X) = \frac{usage(X)}{\sum_{X^* \in CT} usage(X^*)}. \tag{1}$$

An example of code table CT and covering with the patterns from CT is given in Figure 1, d and e, respectively. The computation of CT is guided by the MDL principle – a pattern is added to CT only if it allows for reducing the total description length.

The total length is given by $L(\mathcal{D}, CT) = L(CT|\mathcal{D}) + L(\mathcal{D}|CT)$, where the length of the dataset D encoded by this CT is $L(\mathcal{D}|CT) = \sum_{X \in CT} usage(X) L(code(X))$. The length of CT is given by $L(CT|\mathcal{D}) = \sum_{X \in CT} L(X|ST) + L(code(X))$.

The code table is filled greedily. We explain the principle of covering by means of a small example given in Figure 2. The candidates to optimal patterns (e.g., patterns from Figure 1, b) are arranged in *the standard candidate order*, i.e., by "frequency" $\downarrow$, "$|X|$" $\downarrow$ and lexicographically $\uparrow$, where $\downarrow/\uparrow$ denotes the descending/ascending order, respectively. Then, one-by-one, patterns are added to the code table, where they are arranged in *the standard cover order*, i.e., by "$|X|$" $\downarrow$, "frequency" $\downarrow$, and lexicographically $\uparrow$. Only those patterns that reduce $L(\mathcal{D}, CT)$ are retained in the code table. The steps of building the optimal CT using candidates AB, ABC, AD, BE (the candidates are considered in the standard candidate order) is shown in Figure 2. The first pattern AB is added to the standard code table ST (Figure 2, a), it reduces $L(\mathcal{D}, CT)$, thus the optimal pattern set is $\{AB\}$. Then ABC is added to CT (Figure 2, b), it reduces $L(\mathcal{D}, CT)$ and the updated code table contains $\{ABC, AB\}$. Then AD is added (Figure 2, c), it does not reduce $L(D, CT)$, the optimal pattern set remains unchanged. The same for BE (Figure 2, d), it does not reduce $L(\mathcal{D}, CT)$. For the sake of simplicity, we show only covering by patterns, the corresponding code tables can be reconstructed based on the covering as in the example in Figure 1 (d, e).

3 Keep It Simple: an Algorithm for Discovering Useful Pattern Sets

3.1 Simply-derivable patterns

One of the main difficulties of PM approaches is that they compute candidates to optimal patterns in advance (in contrast to the tree-based approaches, where a tree is built on the fly). Considering all frequent patterns as candidates is infeasible in practice and implies dealing with redundant (similar) patterns. Restricting the set of frequent patterns by increasing the frequency threshold does not solve this problem. Moreover, some interesting and infrequent patterns can be omitted. Using only closed itemsets partially solves the redundancy problem, however, the number of closed itemsets is still exponential.

We propose a deterministic threshold-free approach for building a candidate set comprising of at most a quadratic number of patterns w.r.t. the number of attributes. These patterns are "easily-derivable," since they are computed based on the closure of a union of two attributes, i.e., for a dataset $\mathcal{D}$ over attributes I, the set of candidates is given by $\mathcal{F} = \{cl(\{i_k i_j\}) \mid i_k, i_j \in I, k > j\}$. in what follows we call these patterns *biclosed itemsets*. For a dataset from Figure 1, a, the set of biclosed itemsets is $\{AB, AD, BE, ABC, ABDE, ABCDE\}$. For biclosed itemsets of size at least 3 there are several ways to derive them, e.g., $cl(AC) = cl(BC) = ABD$, $cl(CD) = cl(CE) = ABCDE$. On the one hand, it is easy to reproduce a biclosed itemset – it is enough to know only two attributes. On the other hand, a biclosed itemset is maximal (cannot be extended without reducing the support). The number of biclosed itemsets is bounded by $|I|(|I| - 1)/2$.

3.2 Multi-model Description

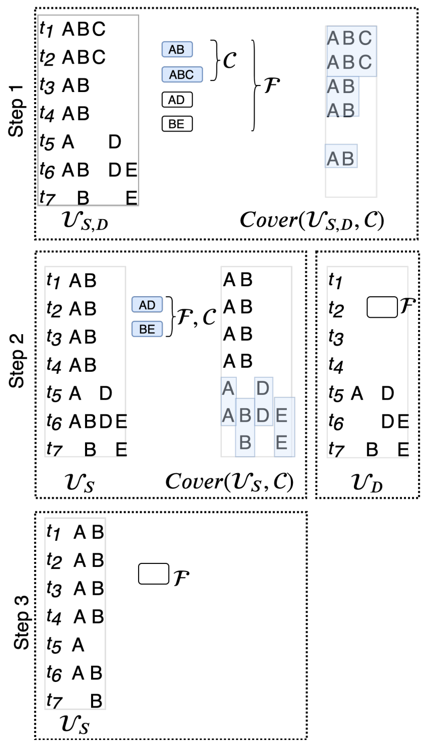
Since the number of biclosed candidates is much lower than the number of frequent patterns, they do not cover a dataset as good as patterns from an exponentially large pattern set, like candidates in Krimp. Thus, larger fragments of data remain uncovered. Using the principle of "boosting", i.e., building a new model based on the data fragments that have been poorly described by the previous models, we propose to explore gradually patterns by a sequence of models (pattern sets). At each iteration, we consider a data fragment (projection) U of the binary dataset $\mathcal{D}$ that has not been entirely described by the models from the previous iterations. We propose an algorithm, called *KeepItSimple* (KIS), the pseudocode is given in Algorithm 1. We consider two versions of the algorithm that differ in the way of computing projections (i.e., defining which fragments are insufficiently well described).

To compute a single model we use Krimp algorithm. However, instead of frequent (closed) patterns, we use biclosed itemsets as candidates. We also change the probability estimates (in Formula 1), we give details in Section 3.3. At the beginning, KIS works as follows: the initial code table is *the standard code table* (line 1), the candidates $\mathcal{F}$ are biclosed itemsets (line 3). Then candidates $\mathcal{F}$ are used to cover greedily the dataset. We use the same cover principles as in Krimp (see Section 2.2 for details). When the first pattern set (model) is computed there are two possibilities to define projections, i.e., the fragments U of data that will be used instead of $\mathcal{D}$ at the next iteration to find a new set of optimal patterns. We consider a *simple* (S) and *detailed* (D) projections. To compute projection (line 12), we consider attributes $I^* \subseteq I$ that are partially covered by patterns from the current code table. To compute S-projection, we keep the whole columns corresponding to partially covered attributes, while to compute D-projection we keep only uncovered fragments in columns (see line 12 S and D, respectively). The process of computing new tables stops when there are no biclosed patterns or the next optimal code table is the standard one.

Example. Let us consider how the algorithms work using the dataset from Figure 1. The steps of Algorithm 1 are given to the right of the pseudocode. At the first step we use the original dataset $\mathcal{D}$ instead of projections. The optimal pattern set is $\mathcal{C} = \{AB, ABC\}$. For covered data $Cover(U_{S,D}, \mathcal{C})$ the projections U_S and U_D are given in Step 2. For S-projection, the whole columns are used, while for D-projection we use only uncovered fragments (U_S and U_D, respectively). These projections are used to compute new candidates. Biclosed itemsets computed on the projected data U_S and U_D are $\mathcal{F} = \{AD, BE\}$ and $\mathcal{F} = \{\}$, respectively. The optimal patterns are $\mathcal{C} = \{AD, BE\}$ and $\mathcal{C} = \emptyset$, for S and D-projections, respectively. Step 3 is performed only for the S-projected data. Since the only possible biclosed itemset AB is already in the set of optimal patterns, the candidate set $\mathcal{F}$ is empty. The optimal pattern sets are $\mathcal{C} = \{AB, ABC, AD, BE\}$ and $\mathcal{C} = \{AB, ABC\}$, for S- and D-projections, respectively.

Algorithm 1 KeepItSimple($\mathcal{D}$)

```
Require: binary dataset $\mathcal{D}$ over set of attributes $I$
Ensure: itemset collection $\mathcal{P}$
 1: $L \leftarrow StandardCodeTableLength(\mathcal{D})$
 2: $U \leftarrow \mathcal{D}$; $I^* \leftarrow I$
3S: $\mathcal{F} \leftarrow ComputeBiclosed(U)$
 4: $run \leftarrow True$
 5: while $run$ do
 6:     $L_{old} \leftarrow L$
7S:     $\mathcal{F} \leftarrow ComputeInducedBiclosed(\mathcal{F}, I^*)$
7D:     $\mathcal{F} \leftarrow ComputeBiclosed(U)$
 8:     $\mathcal{C} \leftarrow MDLOptimal(\mathcal{U}, \mathcal{F})$
 9:     $L \leftarrow L(U \mid \mathcal{C}) + L(\mathcal{C} \mid U)$
10:     $U \leftarrow U \setminus Cover(U, \mathcal{C})$
11:     $I^* \leftarrow UncoveredAttributes(U)$
12S:    $U \leftarrow Projection(\mathcal{D}, I^*)$
12D:    $U \leftarrow Projection(U, I^*)$
13:     $\mathcal{P} \leftarrow \mathcal{P} \cup \mathcal{C}$
14:     $run \leftarrow run$ and $(L_{old} > L)$ and $(I^* \neq \emptyset)$
15: end while
16: return $\mathcal{P}$
```



3.3 Alternative Estimates of Patterns

We also propose alternative estimates of pattern probability, instead of the usage in covering (Formula 1) we propose to use frequency:

$$P(X) = \frac{frequency(X)}{\sum_{X^* \in CT} frequency(X^*)}. \tag{2}$$

Replacing usage by frequency means that patterns may overlap. According to the compression principle, repetitive encoding does not provide the best compression (reduction of $L(D, CT)$ w.r.t. $L(D, ST)$), but imposes a heavier penalty on pattern similarity. We leave the study of the side effects of these changes out of the scope of this paper. In Figure 3 we show how covering changes w.r.t. Krimp (see Figure 2) for the same candidates arranged in the same order. The set of MDL-optimal patterns $\{ABC, AB\}$ computed by Krimp (Figure 2, b) and the set of MDL-optimal patterns $\{AB, AD, BE\}$ computed with the proposed estimates (Figure 3, d) are different. The latter set covers a larger fragment of data and the optimal patterns are less similar.

In our experiments we show that the frequency-based estimate allows one to improve the quality of obtained pattern sets.

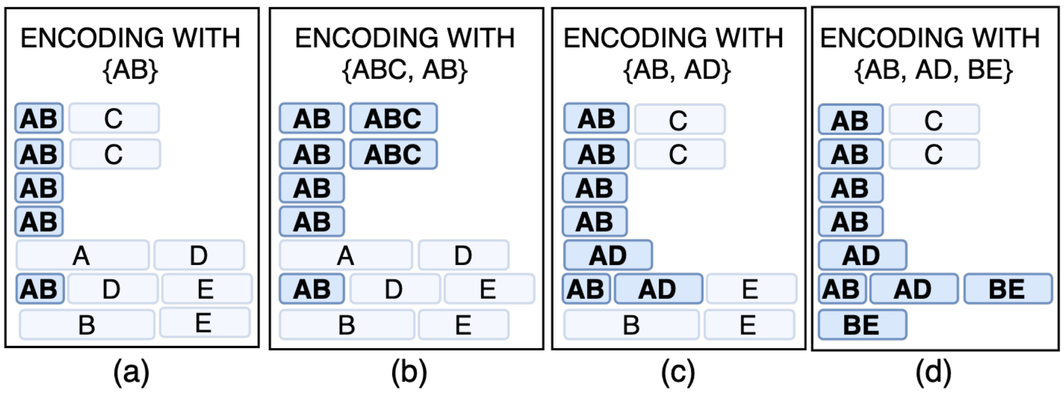


Figure 3: (a) adding AB (optimal), (b) adding ABC (not optimal), (c) adding AD (optimal), (d) adding BE (optimal). The final set of optimal patterns is $\{AB, AD, BE\}$, the optimal encoding is given in (b).

4 Experiments

In this section we report the results of an experimental study. We compare the following algorithms: Krimp [19] (with disjoint estimates, given in Formula 1) and KIS (with disjoint and overlapping estimates, given in Formula 2). For KIS we distinguish 2 types of projections, i.e., simple S and detailed D. We evaluate our methods on freely available datasets (*adult*, *auto*, *breast*, *car evaluation*, *chess*, *ecoli*, *glass*, *heart disease*, *hepatitis*, *iris*, *led 7*, *mushroom*, *nursery*, *pima indians*, *soybean large*, *tic-tac-toe*, *wine*, *zoo*) from the LUCS/KDD discretized data set repository [4].

As quality measures, we consider the following ones.

Size of pattern sets. The pattern sets of a smaller size can be more easily checked by experts. Thus, the smaller sizes are preferable.

Descriptiveness (coverage ratio). We define the coverage ratio as the ratio of entries that are covered by patterns to all non-empty entries. A pattern set is more descriptive if it covers a larger portion of data.

Redundancy (average number of patterns that describe an entry). For non-redundant pattern sets, this value is close to 1. The high values indicate redundancy, i.e., an entry is described by several patterns.

Interestingness (F-measure, precision, recall). We understand interestingness as the ability of patterns to describe "hidden interesting groups of objects". To evaluate interestingness we consider labeled datasets, where class labels were used only to evaluate patterns. As quality measure we choose *F-measure* (F_1 score), as it allows us to take into account both precision and recall of a description:

$$F_1 = 2\frac{precision \cdot recall}{precision + recall}.$$

The results of the experiments are given in Figure 4. The bars/dashes correspond to the averaged values/standard deviation, respectively. Figure 4 shows that KIS provides more stable results, i.e., the standard deviation is smaller for the first three characteristics, namely the number of optimal patterns, coverage ratio and the number of patterns per cell. The quality of patterns (evaluated with $F1$ measure) is higher for KIS (any settings) than for Krimp. Further, we compare approaches in more detail.

Models with disjoint estimates. Let us compare Krimp and KIS based on S- and D-projections. To estimate pattern probability, we use Formula 1 ("disjoint" estimates). Among KIS(S), KIS(D), and Krimp, KIS(S) has the largest average number of patterns (Figure 4, the "disjoint" bar), the highest coverage ratio and the largest number of patterns per cell. That allows us to conclude that KIS(S)-optimal pattern sets are as redundant as Krimp-optimal ones. However, having a larger number of patterns, KIS(S)-optimal pattern sets describe better datasets. KIS(D) generates roughly the same number of patterns as Krimp and has similar redundancy, but allows describing fragments of data larger than those described by Krimp.

From these results we conclude that replacing frequent closed itemsets with biclosed ones and using a multi-model description allows us to improve the descriptiveness (coverage ratio) of pattern sets. In the case of S projections, the complexity of the model (i.e., # patterns, # patterns per cell) does not increase.

Models with overlapping estimates. New probability estimates (in Formula 2) were proposed to improve redundancy of patterns. Let us compare Krimp with KIS algorithm (both simple (S) and detailed (D) projections), where in KIS we use new estimates (Figure 4, "overlapping" bar). The results show that KIS (w.r.t.

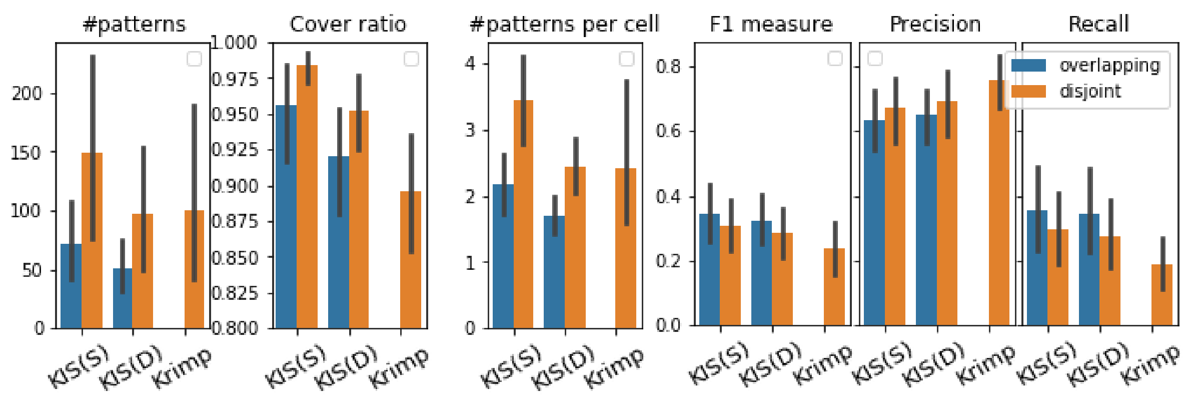


Figure 4: The average values (bars) and standard deviation (dashes) of the quality measures of pattern sets computed by KIS (S, D-projections) and Krimp. For *the number of patterns*, we exclude the results for "adult" dataset since the number of Krimp-generated patterns is drastically higher than the number of KIS (S,D)-generated ones with the same ("disjoint") estimates (1283 vs 639 and 497, respectively).

Krimp) returns smaller pattern sets that describe larger data fragments and have a lower amount of patterns per cell. Thus, using overlapping estimates we get more descriptive models (coverage ratio) that contain less amount of patterns. It is important to notice that the quality of patterns ($F1$-measure) for pattern set computed using overlapping estimates is higher than that for Krimp-optimal.

For KIS we also consider the average number of models computed with two types of estimates. On average, the number of KIS(D)-generated models is lower (2.69 and 2.54 for "disjoint" and "overlapped" estimates, respectively). The average number of KIS(S)-generated models is 4.08 and 3.54 for "disjoint" and "overlapped" estimates, respectively. Thus, KIS based on S-projections converges faster than that based on D-projections, i.e., it needs a lower number of steps to compute the whole set patterns.

To sum up, the proposed modifications, namely (i) (simply-derivable) biclosed itemsets, (ii) multi-model description, and (iii) new probability estimates allow us to improve the results of PM w.r.t. Krimp. Depending on chosen settings we can significantly improve a particular quality measure. The best overall results are achieved for KIS based on D-projections with the proposed (overlapping) estimates. Applying KIS with these settings allows us to reduce considerably the size of pattern sets and pattern redundancy, improving descriptiveness and quality of patterns.

5 Conclusion

In this paper, we have proposed an MDL-based multi-model approach to PM. It is based on an efficient method for computing candidates and combines the best practices of building tree-based supervised models with the state-of-the-art approach called Krimp. The results of experiments show that the proposed approach allows one to obtain non-redundant, interesting pattern sets that describe a large portion of data.

Acknowledgments

The work of Sergei O. Kuznetsov presented in Sections 2-3 was carried out at St. Petersburg Department of Steklov Mathematical Institute of Russian Academy of Science and supported by the Russian Science Foundation grant no. 17-11-01276.

References

[1] Charu C. Aggarwal and Jiawei Han, *Frequent pattern mining*, Springer, 2014.

[2] Mario Boley, Claudio Lucchese, Daniel Paurat, and Thomas Gärtner, 'Direct local pattern sampling by efficient two-step random procedures', in *Proceedings of the 17th ACM SIGKDD DMCD*, pp. 582–590, (2011).

[3] Leo Breiman, 'Random forests', *Machine learning*, **45**(1), 5–32, (2001).

[4] F Coenen, 'The LUCS-KDD discretised/normalised ARM and CARM data library'. Department of CS, The University of Liverpool, UK, (2003).

[5] Diane J Cook and Lawrence B Holder, 'Substructure discovery using minimum description length and background knowledge', *Journal of Artificial Intelligence Research*, **1**, 231–255, (1993).

[6] Thomas M Cover and Joy A Thomas, *Elements of information theory*, John Wiley & Sons, 2012.

[7] Vladimir Dzyuba, Matthijs van Leeuwen, and Luc De Raedt, 'Flexible constrained sampling with guarantees for pattern mining', *Data Mining and Knowledge Discovery*, **31**(5), 1266–1293, (2017).

[8] Yoav Freund and Robert E Schapire, 'A desicion-theoretic generalization of on-line learning and an application to boosting', in *European conference on computational learning theory*, pp. 23–37. Springer, (1995).

[9] Liqiang Geng and Howard J Hamilton, 'Interestingness measures for data mining: A survey', *ACM Computing Surveys (CSUR)*, **38**(3), 9, (2006).

[10] Pierre Geurts, Damien Ernst, and Louis Wehenkel, 'Extremely randomized trees', *Machine learning*, **63**(1), 3–42, (2006).

[11] Peter Grünwald, *The minimum description length principle*, MIT, 2007.

[12] Michael Mampaey, Jilles Vreeken, and Nikolaj Tatti, 'Summarizing data succinctly with the most informative itemsets', *ACM (TKDD)*, **6**(4), 16, (2012).

[13] Nicolas Pasquier, Yves Bastide, Rafik Taouil, and Lotfi Lakhal, 'Discovering frequent closed itemsets for association rules', in *International Conference on Database Theory*, pp. 398–416. Springer, (1999).

[14] Nicolas Pasquier, Yves Bastide, Rafik Taouil, and Lotfi Lakhal, 'Efficient mining of association rules using closed itemset lattices', *Information systems*, **24**(1), 25–46, (1999).

[15] J. Ross Quinlan, 'Induction of decision trees', *Machine learning*, **1**(1), 81–106, (1986).

[16] Arno Siebes and René Kersten, 'A structure function for transaction data', in *Proceedings of SIAM*, pp. 558–569, (2011).

[17] Koen Smets and Jilles Vreeken, 'Slim: Directly mining descriptive patterns', in *Proceedings of SIAM*, pp. 236–247, (2012).

[18] Jilles Vreeken and Nikolaj Tatti, 'Interesting patterns', in *Frequent Pattern Mining*, eds., Charu C. Aggarwal and Jiawei Han, 105–134, Springer, (2014).

[19] Jilles Vreeken, Matthijs Van Leeuwen, and Arno Siebes, 'Krimp: mining itemsets that compress', *Data Mining and Knowledge Discovery*, **23**(1), 169–214, (2011).