ITAT 2016 Proceedings, CEUR Workshop Proceedings Vol. 1649, pp. 147–154
http://ceur-ws.org/Vol-1649, Series ISSN 1613-0073, © 2016 J. Outrata, M. Trnecka

Evaluating Association Rules in Boolean Matrix Factorization

Jan Outrata and Martin Trnecka*

Department of Computer Science
Palacký University Olomouc, Czech Republic
jan.outrata@upol.cz, martin.trnecka@gmail.com

Abstract: Association rules, or association rule mining, is a well-established and popular method of data mining and machine learning successfully applied in many different areas since mid-nineties. Association rules form a ground of the Asso algorithm for discovery of the first (presumably most important) factors in Boolean matrix factorization. In Asso, the confidence parameter of association rules heavily influences the quality of factorization. However, association rules, in a more general form, appear already in GUHA, a knowledge discovery method developed since mid-sixties. In the paper, we evaluate the use of various (other) types of association rules from GUHA in Asso and, from the other side, a possible utilization of (particular) association rules in other Boolean matrix factorization algorithms not based on the rules. We compare the quality of factorization produced by the modified algorithms with those produced by the original algorithms.

1 Introduction

1.1 The problem and algorithms

Boolean matrix factorization (BMF), called also Boolean matrix decomposition, aims to find for a given $n \times m$ object-attribute incidence Boolean matrix I a $n \times k$ object-factor Boolean matrix A and a $k \times m$ factor-attribute Boolean matrix B such that the Boolean matrix product (see below) of A and B (approximately) equals I. A decomposition of I into A and B may be interpreted as a discovery of k factors exactly or approximately explaining the data. Interpreting the matrices I, A and B as the object-attribute, object-factor and factor-attribute incidence matrices, respectively, A and B explain I as follows: the object i has the attribute j (the entry I_{ij} corresponding to the row i and the column j is 1) if and only if there exists factor l such that l applies to i and j is one of the particular manifestations of l. The optimization version of this basic BMF problem demands k to be as small as possible. The least k for which an exact decomposition of I exists is called the *Boolean rank* (or Schein rank) of I.

In the literature, two common variants of the optimization problem are defined and dealt with: the approximate factorization problem (AFP) [3], where the number k of factors is sought to be minimal for a prescribed maximal difference (error) of the Boolean product of A and B from I (usualy zero), and the discrete basic problem (DBP) [11], where the error is sought to be minimal for a prescribed (fixed) k. The formal definitions of the problems are given below. For the former problem, greedy approach algorithms seem popular. One of the most efficient ones, GreConD, is proposed in [3] (where it is called Algorithm 2). Another one, GreEss [4], refines GreConD based on a deeper insight into the problem. For the latter problem, probably the most known (and a basic one) algorithm is Asso, proposed in [11], together with a few of its variants. Other BMF algorithms proposed in the recent data mining literature that can be tailored for either of the two problems are e.g. Hyper [16] or PaNDa [10].

GreConD (and GreEss and their variants) finds factors as *maximal rectangles*, Boolean matrices whose entries with 1 form a maximal rectangular area (full of 1s), upon a suitable permutation of rows and columns. This concept comes from the geometric view on BMF and *formal concept analysis* (FCA). The view tells us that finding a decomposition of I means finding a *coverage* of 1s in I by rectangles full of 1s [3, 4] and in FCA maximal rectangles full of 1s correspond to so-called formal concepts [5]. We will use maximal rectangles as factors to describe the algorithms now. The GreConD algorithm, in its seek for a factor, starts with the empty set of attributes to which a selected attribute with possibly other attributes are repeatedly added. The constructed set of attributes together with the set of all objects having all the attributes determine a maximal rectangle (form a formal concept). The selected attribute is such that the rectangle with the attributes added *covers* as many still *uncovered* 1 in I as possible. The attributes are added repeatedly as long as the number of still uncovered 1s in I covered by the rectangle grows (note the greedy approach here). Further factors are sought the same way and the algorithm stops when the prescribed number of 1s in I is covered by the maximal rectangles corresponding to factors (i.e. the algorithm is designed for the AFP). Characteristic vectors of object sets determining found maximal rectangles then constitute columns of matrix A and characteristic vectors of attribute sets of the maximal rectangles constitute rows of matrix B.

Asso, on the other hand, starts in its seek for each of (at most) the prescribed number k of factors with a selected set of attributes and searches for a set of objects such that the Boolean product of the characteristic vectors of the two sets, as a rectangle (not necessarily maximal) correspond-

*Support by grant No. GA15-17899S of the Czech Science Foundation is acknowledged. M. Trnecka also acknowledges partial support by grant No. PrF_2016_027 of IGA of Palacký University Olomouc.

ing to the factor, covers as many 1s in I as possible. At the same time, however, the rectangle must also *overcover* as few 0 in I by 1 as possible (i.e. the algorithm is designed for the DBP). Note here that ASSO is admitted to overcover 0 in I in its aim to cover 1s, while GRECOND does not do that (thereby it is said it performs a from-bellow decomposition of I). Characteristic vectors of the selected sets of attributes constitute rows of matrix B while characteristic vectors of the corresponding found sets of objects constitute columns of matrix A. The sets of attributes are selected, under the same conditions as for the search of objects, among candidate sets represented by rows of the attribute-attribute matrix called association matrix. This $m \times m$ Boolean matrix, created in the first step of the algorithm, contains 1 in the row corresponding to an attribute i and the column corresponding to an attribute j if the association rule $i \Rightarrow j$ has the (so-called) confidence – the ratio of the number of objects having both i and j to the number of objects having i – greater than a user-defined parameter τ; otherwise it contains 0 in the row and the column.

1.2 Associations in BMF

The association rules in ASSO, also known from association rule mining [1], are of one type of relationship between two (Boolean) attributes in (Boolean) object-attribute data. There are other types, in general between two logical formulas above attributes instead of between just two single attributes, introduced in the literature. After all, general association rules whose validity in data is defined through a function of the numbers of objects having and not having both or one of the attributes (or, in general, satisfying and not satisfying the formulas above attributes) are introduced and operated with in GUHA [7, 8, 9, 13], one of the oldest, less known but most sophisticated, method of knowledge discovery. In GUHA, several logical and statistical functions, called quantifiers, used to interpret several different types of association rules are introduced. Actually, one of the quantifiers (the basic one), founded implication, interprets the association rule used in ASSO (and association rule mining, see below).

The topic of this paper is twofold. First, we pick up several (other) types of association rules, interpreted by selected GUHA quantifiers, and use them in place of the ASSO's association rule in the ASSO algorithm. Second, vice versa, we take the concept of association matrix from ASSO and utilize it in greedy-approach algorithms. Namely, we take a particular association matrix and use the rows of the matrix as characteristic vectors of candidates to initial sets of attributes to which further attributes are added in GRECOND instead of the empty set. Both modifications of the algorithms are novel ideas not previously discussed in the literature. The main purpose of the paper is to evaluate the use of various types of association rules in ASSO algorithm and the use of association matrix in GRECOND algorithm. The evaluation is done by experimental comparison of quality of decompositions obtained from the modified algorithms with those obtained from their respective original versions.

The rest of the paper is organized as follows. In the following section 2 we briefly precise the BMF problems introduced above and recall GUHA, namely the quantifiers and general association rules. Then, in section 3 the modification of ASSO with GUHA association rules and the modification of GRECOND with rows of particular association matrix as initial attributes sets are presented. The modified algorithms are experimentally evaluated in section 4 and section 5 draws a conclusion and future research directions.

2 Basic notions of BMF and GUHA

2.1 Boolean Matrix Factorization

We precise the basic BMF notions and problems recalled in the beginning of the previous section. Denote by $\{0,1\}^{n\times m}$ the set of all $n \times m$ Boolean matrices (i.e. with entries either 1 or 0). The basic problem in BMF is to find for a given $I \in \{0,1\}^{n\times m}$ matrices $A \in \{0,1\}^{n\times k}$ and $B \in \{0,1\}^{k\times m}$ for which

$$I \text{ (approximately) equals } A \circ B, \tag{1}$$

where $\circ$ is the Boolean matrix product, i.e.

$$(A \circ B)_{ij} = \max_{l=1}^{k} \min(A_{il}, B_{lj}).$$

The approximate equality in (1) is assessed by means of the well-known L_1-norm $\|I\| = \sum_{i,j=1}^{m,n} |I_{ij}|$ and the corresponding distance (error) function $E(I, A \circ B)$ defined by

$$E(I, A \circ B) = \|I - A \circ B\| = \sum_{i,j=1}^{m,n} |I_{ij} - (A \circ B)_{ij}|.$$

In BMF, one is usually interested in two parts of the error function E, E_u corresponding to 1s in I that are 0s (and hence *uncovered*) in $A \circ B$ and E_o corresponding to 0s in I that are 1s (and hence *overcovered*) in $A \circ B$:

$$\begin{aligned} E(I, A \circ B) &= E_u(I, A \circ B) + E_o(I, A \circ B), \text{ where} \\ E_u(I, A \circ B) &= |\{\langle i, j\rangle ; I_{ij} = 1, (A \circ B)_{ij} = 0\}|, \\ E_o(I, A \circ B) &= |\{\langle i, j\rangle ; I_{ij} = 0, (A \circ B)_{ij} = 1\}|, \end{aligned}$$

or, more often, in the relative errors

$$e_u(l) = E_u(I, A \circ B)/\|I\|, \quad e_o(l) = E_o(I, A \circ B)/\|I\|. \tag{2}$$

e_u and e_o are functions of the first l factors delivered by the particular algorithm and measure how well the data is explained by the l factors. The value of $1 - e_u$ represents the (pure) coverage of data by the observed factors. We will use $1 - e_u$ and e_o in the experiments section 4 below. Note that the value of $c = 1 - e_u - e_o$ represents the overall coverage of data by the factors and is commonly used to assess quality of decompositions delivered by BMF algorithms [3, 4, 6, 11].

The two optimization BMF problems are defined as follows:

Definition 1 (Approximate Factorization Problem, AFP [3])**.** *Given $I \in \{0,1\}^{n\times m}$ and prescribed error $\varepsilon \geq 0$, find $A \in \{0,1\}^{n\times k}$ and $B \in \{0,1\}^{k\times m}$ with k as small as possible such that $\|I - A \circ B\| \leq \varepsilon$.*

AFP emphasizes the need to account for (and thus to explain) a prescribed (presumably reasonably large) portion of data, which is specified by ε.

Definition 2 (Discrete Basis Problem, DBP [11])**.** *Given $I \in \{0,1\}^{n\times m}$ and a positive integer k, find $A \in \{0,1\}^{n\times k}$ and $B \in \{0,1\}^{k\times m}$ that minimize $\|I - A \circ B\|$.*

DBP emphasizes the importance of the first k (presumably most important) factors. A throughout study of Boolean matrix factorization from the point of views of the two problems is provided in [4].

2.2 GUHA

We will only recall the necessary notions from the GUHA (General Unary Hypotheses Automaton) method [7, 8] which are required to describe the various types of association rules used in the modified ASSO algorithm. For throughout treatise of the foundations of the method (of mechanized formation of hypotheses, in particular its observational predicate calculi), see books [9] or [13].

GUHA, more precisely its ASSOC procedure [9] (do not confuse with the ASSO algorithm!) or more enhanced 4FT-MINER procedure [14] for Boolean data, inputs (Boolean) object-attribute incidence data with Boolean attributes which we represent by a $n \times m$ Boolean matrix I. (General) association rule (over a given set of attributes) is an expression

$$i \approx j$$

where i and j are attributes. (Note that in its full form, GUHA general association rule is an expression $\varphi \approx \psi$ where φ and ψ are arbitrary complex logical formulas above the attributes. We consider the simplified case with just single attributes for the formulas, as in the association rules used in ASSO and association rule mining.) $i \approx j$ is true in I if there is an association between i and j interpreted by a function q assigning to the four-fold table 4ft(i, j, I) corresponding to $i \approx j$ and I the value 1 (logical true). 4ft(i, j, I) is the quadruple

$$\langle a,b,c,d\rangle = \langle fr(i\wedge j), fr(i\wedge\neg j), fr(\neg i\wedge j), fr(\neg i\wedge\neg j)\rangle$$

where $fr(i\wedge j)$ is the number of objects having both i and j in I (rows in I in which there is 1 in both columns corresponding to i and j) and $\neg i$ is an attribute corresponding to the negation of attribute i (i.e. the column in I corresponding to i in which 1s are replaced by 0s and vice versa).

4ft(i, j, I) is usually depicted as

I	j	$\neg j$
i	$a = fr(i\wedge j)$	$b = fr(i\wedge\neg j)$
$\neg i$	$c = fr(\neg i\wedge j)$	$d = fr(\neg i\wedge\neg j)$.

Function q which assigns to any four-fold table 4ft(i, j, I) a logical value 0 or 1 defines a so-called (generalized, GUHA) quantifier. There are several different quantifiers, summarized e.g. in [13], logical and statistical, which interpret different types of association rules (with different meaning of the association $\approx$ between attributes):

- *founded (p-)implication*, $\Rightarrow_p$ (for $\approx$)

$$q(a,b,c,d) = \begin{cases} 1 \text{ if } \frac{a}{a+b} \geq p, \\ 0 \text{ otherwise.} \end{cases}$$

  Parameter $0 < p \leq 1$ has a meaning of threshold for the confidence of the association rule $i \Rightarrow_p j$, i.e. the ratio of the number of objects having in I both attributes i and j to the number of objects having i. Founded implication interprets the association rule used in the original ASSO algorithm (with p denoted as τ instead) and association rule mining, which is thus a particular case of GUHA general association rules.

- *double founded implication*, $\Leftrightarrow_p$

$$q(a,b,c,d) = \begin{cases} 1 \text{ if } \frac{a}{a+b+c} \geq p, \\ 0 \text{ otherwise.} \end{cases}$$

  Compared to founded implication, double founded implication, to evaluate to 1, requires that the number of objects having in I both i and j is at least $100\cdot p\,\%$ of the number of objects having *i or j*.

- *founded equivalence*, $\equiv_p$

$$q(a,b,c,d) = \begin{cases} 1 \text{ if } \frac{a+d}{a+b+c+d} \geq p, \\ 0 \text{ otherwise.} \end{cases}$$

  Meaning: At least $100\cdot p\,\%$ among all objects in I have the same attributes.

- *E-equivalence*, $\sim^E_\delta$

$$q(a,b,c,d) = \begin{cases} 1 \text{ if } \max\left(\frac{b}{a+b}, \frac{c}{c+d}\right) < \delta, \\ 0 \text{ otherwise.} \end{cases}$$

  Meaning: Less than $100\cdot\delta\,\%$ ($0 < \delta \leq 0.5$) of objects among the objects having i do not have j and among the objects not having i have j.

- *negative Jaccard distance*

$$q(a,b,c,d) = \begin{cases} 1 \text{ if } \frac{b+c}{b+c+d} \geq p, \\ 0 \text{ otherwise.} \end{cases}$$

  This is our new quantifier resembling Jaccard distance dissimilarity measure used in data mining (which is one minus Jaccard similarity coefficient [15] which in turn is equal to double founded implication above). Compared to double founded implication, this quantifier, to evaluate to 1, requires that at least $100\cdot p\,\%$ objects have *i or j* among the objects *not* having i or j.

In fact, the above presented quantifiers, except for the last one, are simplified versions of quantifiers defined in [9] where additional parameter $s > 0$ is included:

- *founded implication,* $\Rightarrow_{p,s}$

$$q(a,b,c,d) = \begin{cases} 1 \text{ if } \frac{a}{a+b} \geq p \text{ and } a \geq s, \\ 0 \text{ otherwise.} \end{cases}$$

and similarly for the other quantifiers. For association rule $i \approx j$, s has a meaning of threshold for the so-called support of the rule – the number of objects having in I both attributes i and j (or, if normalized as in association rule mining, the ratio of the number to the number of all objects in I).

3 The modified algorithms

3.1 ASSO with GUHA association rules

The modified ASSO algorithm involving GUHA (general) association rule interpreted by a GUHA quantifier is depicted in Algorithm 1.

Algorithm 1: Modified ASSO algorithm

Input: A Boolean matrix $I \in \{0,1\}^{n\times m}$, a positive integer k, a threshold value $\tau \in (0,1]$, real-valued weights w^+, w^- and a quantifier q_τ (with parameter τ) interpreting $i \approx j$

Output: Boolean matrices $A \in \{0,1\}^{n\times k}$ and $B \in \{0,1\}^{k\times m}$

1 **for** $i = 1,\ldots,m$ **do**
2 **for** $j = 1,\ldots,m$ **do**
3 $Q_{ij} = q_\tau(a,b,c,d)$
4 **end**
5 **end**
6 $A \leftarrow$ empty $n \times k$ Boolean matrix
7 $B \leftarrow$ empty $k \times m$ Boolean matrix
8 **for** $l = 1,\ldots,k$ **do**
9 $(Q_{i\_}, e) \leftarrow \arg\max_{Q_{i\_},\, e\in\{0,1\}^{n\times 1}}$ $cover(\begin{bmatrix} B \\ Q_{i\_} \end{bmatrix}, [A\ e], I, w^+, w^-)$
10 $A \leftarrow [A\ e]$, $B \leftarrow \begin{bmatrix} B \\ Q_{i\_} \end{bmatrix}$
11 **end**
12 **return** A and B

$Q_{i\_}$ denotes the ith row vector of the (Boolean) association matrix Q and the function $cover(B,A,I,w^+,w^-)$ is defined as

$$w^+|\{\langle i,j\rangle ; I_{ij} = 1, (A\circ B)_{ij} = 1\}| \\ -w^-|\{\langle i,j\rangle ; I_{ij} = 0, (A\circ B)_{ij} = 1\}|.$$

The original algorithm was described in the introduction section 1. The only modification in Algorithm 1 to the (generic) version of the original algorithm in [11] is computing the association matrix Q (lines 1–5) using the given quantifier $q_\tau(a,b,c,d)$ interpreting a (general) association rule $i \approx j$ instead of using the confidence of the association rule $i \Rightarrow_\tau j$.

3.2 GRECOND using association matrix

Due to the particular way of finding factors in the GRECOND algorithm, namely as maximal rectangles, we need to use a particular association matrix of which the rows are used as characteristic vectors of candidates to initial attribute sets in the algorithm. The matrix used is computed using the GUHA quantifier founded implication with parameter p set to 1; hence the confidence of the interpreted association rule $i \Rightarrow_p j$ must be 1 for the rule to be true (which precisely coincides with the notion of attribute implication between attributes i and j in formal concept analysis, see [5]). This, at the same time, means that the association matrix is the special case of the association matrix of the ASSO algorithm with $\tau = 1$.

The modified GRECOND algorithm using the association matrix is depicted in Algorithm 2.

Algorithm 2: Modified GRECOND algorithm

Input: A Boolean matrix $I \in \{0,1\}^{n\times m}$ and a prescribed error $\varepsilon \geq 0$

Output: Boolean matrices $A \in \{0,1\}^{n\times k}$ and $B \in \{0,1\}^{k\times m}$

1 $Q \leftarrow$ empty $m \times m$ Boolean matrix
2 **for** $i = 1,\ldots,m$ **do**
3 **for** $j = 1,\ldots,m$ **do**
4 **if** $i \Rightarrow_1 j$ is *true* in I **then**
5 $Q_{ij} = 1$
6 **end**
7 **end**
8 **end**
9 $A \leftarrow$ empty $n \times k$ Boolean matrix
10 $B \leftarrow$ empty $k \times m$ Boolean matrix
11 **while** $\|I - A\circ B\| > \varepsilon$ **do**
12 $D \leftarrow \arg\max_{Q_{i\_}} cover(Q_{i\_}, I, A, B)$
13 $V \leftarrow cover(D, I, A, B)$
14 **while** there is j such that $D_j = 0$ and $cover(D + [j], I, A, B) > V$ **do**
15 $j \leftarrow \arg\max_{j, D_j=0} cover(D + [j], I, A, B)$
16 $D \leftarrow (D + [j])^{\downarrow\uparrow}$
17 $V \leftarrow cover(D, I, A, B)$
18 **end**
19 $A \leftarrow [A\ D^{\downarrow}]$, $B \leftarrow \begin{bmatrix} B \\ D \end{bmatrix}$
20 **end**
21 **return** A and B

D_j denotes the jth item of (row Boolean) vector $D \in \{0,1\}^{1\times m}$, $[j] \in \{0,1\}^{1\times m}$ denotes the (row Boolean) vec-

Dataset	k	dens A	dens B	dens I
Set C1	40	0.07	0.04	0.10
Set C2	40	0.07	0.06	0.15
Set C3	40	0.11	0.05	0.20

Table 1: Synthetic data

Dataset	Size	$\|\|I\|\|$
DNA	4590×392	26527
Mushroom	8124×119	186852
Zoo	101×28	862

Table 2: Real data

tor with jth item equal to 1 and all other items equal to 0, the function $cover(D,I,A,B)$ is defined as

$$\|(D^{\downarrow} \times D^{\downarrow\uparrow}) \cdot (I - A \circ B)\|$$

and the (formal concept-forming [5]) operators $C^{\uparrow}$ and $D^{\downarrow}$ for (column Boolean) vector $C \in \{0,1\}^{n\times 1}$ and vector D, respectively, are defined as

$$C^{\uparrow} = \bigwedge [j] \in \{0,1\}^{1\times m};\ \text{for each } i, C_i = 1 : I_{ij} = 1,$$
$$D^{\downarrow} = \bigwedge [i] \in \{0,1\}^{n\times 1};\ \text{for each } j, D_j = 1 : I_{ij} = 1.$$

Again, the original algorithm was described in the introduction section 1. The only modifications in Algorithm 2 to the (generic) version of the original algorithm in [3] are computing the particular association matrix Q (lines 1–8) using the quantifier founded implication with $p = 1$ interpreting the association rule $i \Rightarrow_1 j$ and using the rows of the matrix as characteristic vectors of candidates to initial attribute sets (line 12) in the factor construction (lines 12–19).

4 Experimental evaluation

In this section, we provide an experimental evaluation of the modified algorithms and their comparison to the original versions, the ASSO algorithm and the GRECOND algorithm. Due to the lack of space we do not present a comparison with other algorithms and approaches to the general BMF. A comparison that includes both the algorithms can be found for example in [4].

As in the typical experiment scenario—which occurs in various BMF papers—we use both synthetic and real datasets. Experiments on synthetic datasets enable us to analyze performance of the algorithms on data with the same and known characteristics—we can analyze results in average case. On real data, we can study meaning of obtained results. Let us also note, that synthetic data are artificial while real data are influenced by real factors.

Synthetic data. We created 1000 of randonly generated datasets. Every dataset X_i has 500 rows and 250 columns and was obtained as a Boolean product $X_i = A_i \circ B_i$ of matrices A_i and B_i that were generated randomly with parameters shown in Table 1. The inner dimmension k was for all X_i set to 40, i.e. the expected number of factors is 40.

Real data. We used datasets DNA [12], Mushroom [2], and Zoo [2], the characteristics of which are shown in Table2. All of them are well known and widely used in the literature on BMF.

4.1 ASSO with GUHA association rules

We observe the values of $1 - e_u(l)$ (2) for $l = 0,\ldots,k$ where k is the number of factors delivered by a particular algorithm. Clearly, for $l = 0$ (no factors, A and B are "empty") we have $1 - e_u(l) = 0$. In accordance with general requirements on BMF, for a good factorization algorithm $1 - e_u(l)$ should be increasing in l, should have relatively large values for small l, and it is desirable that for $l = k$ we have $I = A \circ B$, i.e. the data is fully explained by all k factors computed (in which case $1 - e_u(l) = 1$). For synthetic datasets C_1, C_2 and C_3, values of $1 - e_u(l)$ are shown in Figures 1, 2 and 3, respectively.

As we mentioned above the ASSO algorithm admits overcovering of 0s of input data matrix. The number of overcovered 0s is a very important value and the values of $e_o(l)$ (2) for synthetic datasets C_1, C_2 and C_3 are shown in Figures4, 5 and 6. Let us note that the results marked as "founded implication" are in fact results for the original ASSO algorithm. Note also that all variants of ASSO require us to set τ and (one of) $w+$ and $w-$, see Algorithm 1. Based on some previous experimental results (see [4]) we fixed $w+ = w- = 1$ and used the value of τ for which the particular algorithm produces the best results. In most cases, the best choice was $0.8 < \tau < 1$. This observation corresponds with results in [4].

We can see that the original algorithm is outperformed in terms of both coverage $(1 - e_u)$ and overcoverage (e_o) by the modification utilizing double founded implication. This modification produces large values of coverage and compared to the original ASSO algorithm commits smaller overcoverage error. This is true for both synthetic and real datasets. Very promising is also the modification utilizing the negative Jaccard distance quantifier.

Modifications utilizing founded equivalence and E-equivalence, however, do not perform well, for synthetic datasets. In case of dataset C_1—the most sparse one—both modifications commit extremely large overcover error, the values are beyond the scale of Figure 4. In cases of C_2 and C_3, Figures 2 and 3, they produce poor coverage while the overcoverage error is not much different from the modifications utilizing other quantifiers, for higher number of factors. On the other side, for real datasets the results are comparable with the other modifications (Figures 7, 8), with significantly smaller overcoverage errors (Figures 10, 11). The only exception is the Mushroom dataset where founded equivalence is again beyond the scale of Figure 10. Due to the limited space we do not include results for the DNA dataset which are very close to the results obtained for the Zoo dataset.

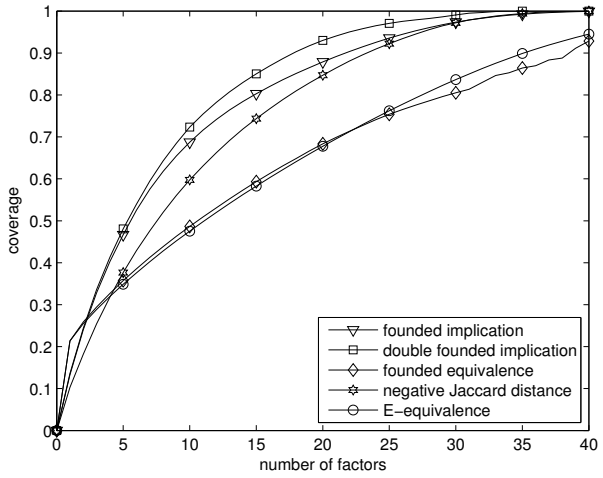

Figure 1: Coverage for synthetic dataset C_1

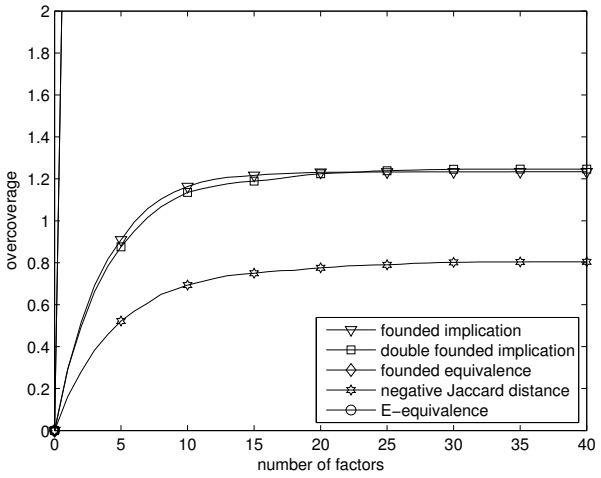

Figure 4: Overcoverage for synthetic dataset C_1

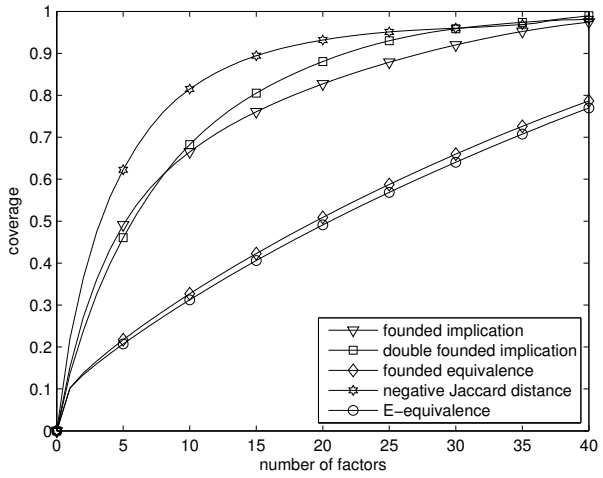

Figure 2: Coverage for synthetic dataset C_2

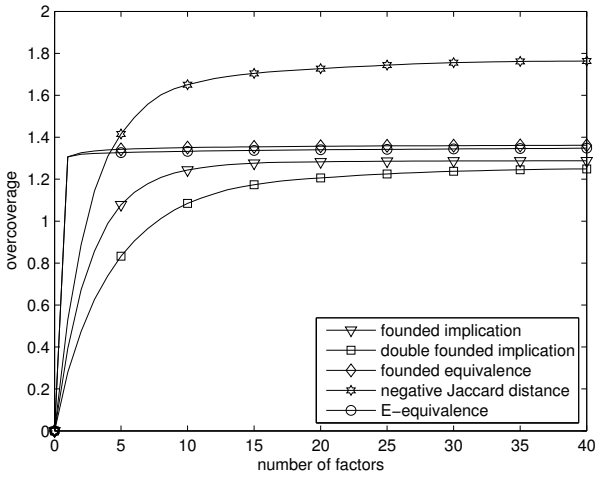

Figure 5: Overcoverage for synthetic dataset C_2

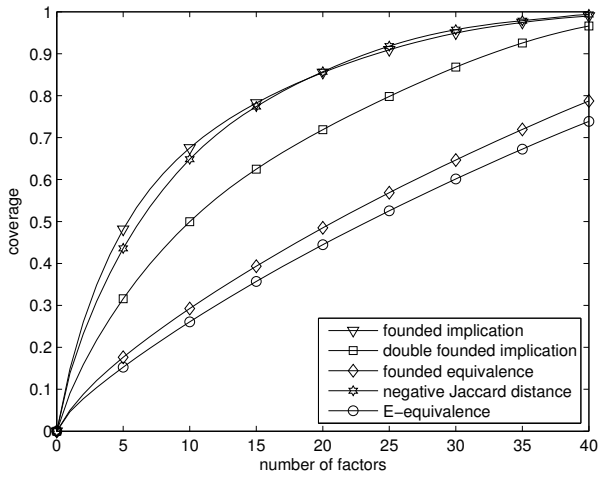

Figure 3: Coverage for synthetic dataset C_3

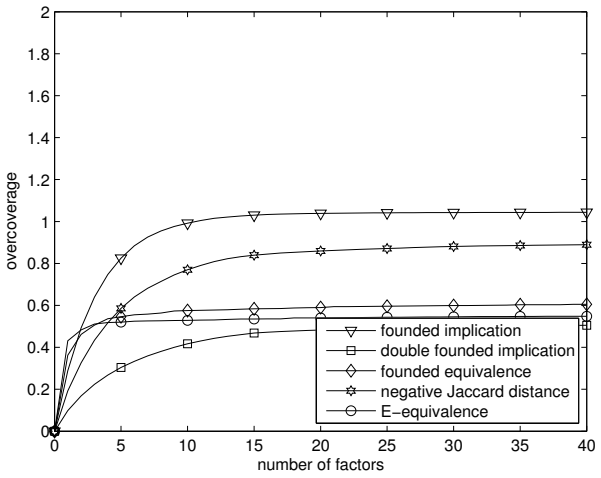

Figure 6: Overcoverage for synthetic dataset C_3

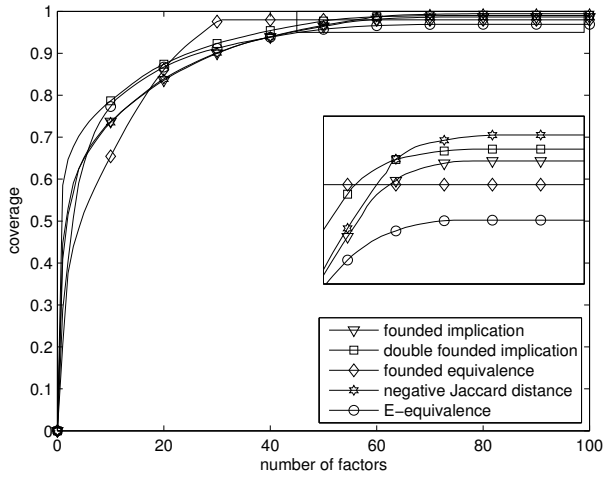

Figure 7: Coverage for Mushroom dataset

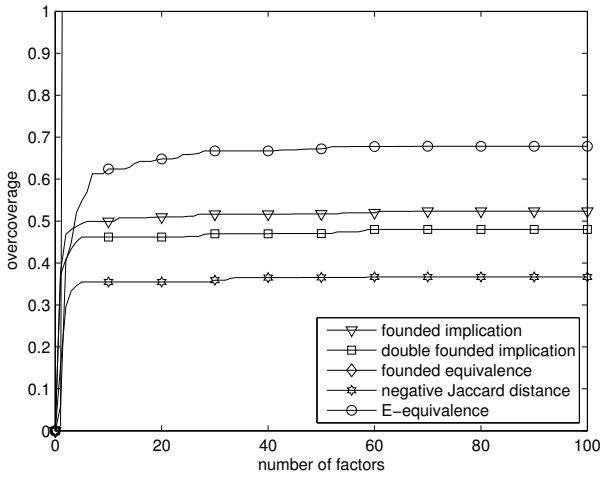

Figure 10: Overcoverage for Mushroom dataset

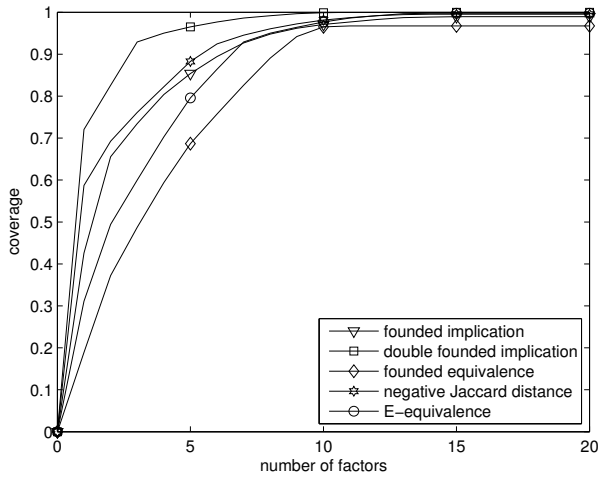

Figure 8: Coverage for Zoo dataset

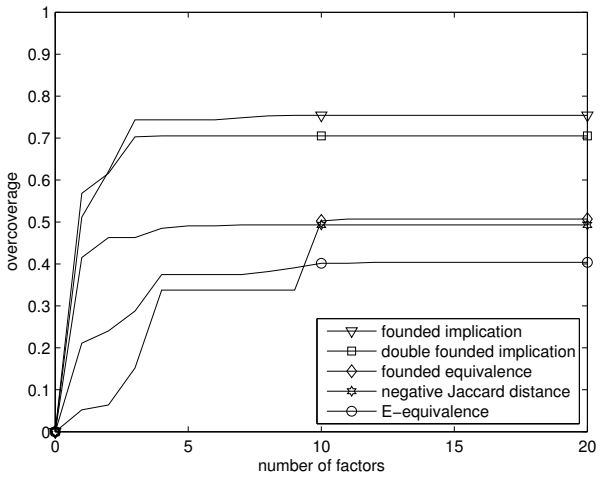

Figure 11: Overcoverage for Zoo dataset

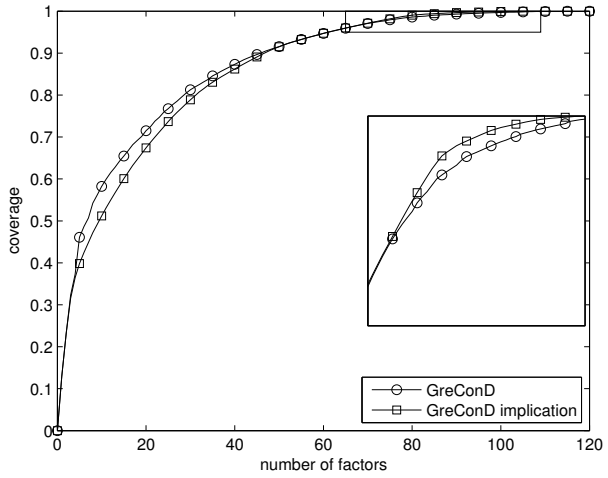

Figure 9: Original and modified GRECOND on Mushroom dataset

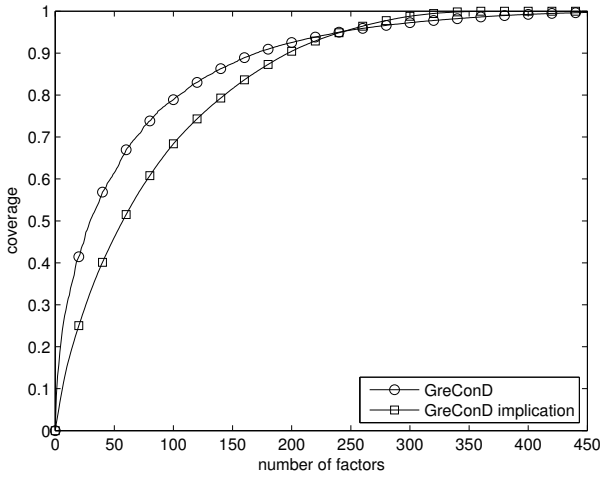

Figure 12: Original and modified GRECOND on DNA dataset

Presented results are representative. We performed the same evaluations for several other datasets which we have not included in the paper and observed the same type of results described above.

4.2 GreConD using association matrix

Do to the limited space we do not present a comparison of the original GreConD algorithm and the modification utilizing the association matrix (Algorithm 2) on the synthetic datasets. On each X_i the modified GreConD slightly outperforms the original algorithm from the standpoint of coverage. Moreover, the modified algorithm also tends to produce less factors, i.e. outperforms the original GreConD from both the AFP and DBP views (see Section 2).

Values of $1 - e_u$ for the Mushroom and DNA datasets are shown in Figures 9 and 12, respectively. We can see that the modified algorithm first looses for small numbers of factors but in the end, it outperforms the original GreConD algorithm—i.e. it outperforms GreConD from the AFP view. We observed similar behavior also for the other real datasets.

Time complexity. We implemented all used algorithms in MATLAB with critical parts written in C. Theoretical time complexity is not of our primary concern. Practically, it follows from our observations that the modification of the GreConD algorithm is slightly faster than the original algorithm and all modifications of the Asso algorithm are equally fast as the original.

5 Conclusion

We evaluated the use of various types of (general) association rules from the GUHA knowledge discovery method in the Boolean matrix factorization (BMF) algorithm Asso and an utilization of (particular) association rules in the greedy-approach BMF algorithm GreConD which is not based on association rules. The comparison of the quality of factorization produced by the modified algorithms with those produced by the original algorithms on both synthetic and selected real data showed that our modified algorithms outperform, for some types of rules, the original ones. Namely the double founded implication and (our new) negative Jaccard distance quantifiers interpreting the association rules in Asso perform better than the founded implication quantifier used in the original Asso. Also the utilization of association matrix in the factor search initialization stage of the GreConD algorithm improves factorization results produced by the algorithm.

The observed results encourage us to the following future research directions. First, as the role of association matrix is crucial for the Asso algorithm (and, as we have seen, the algorithm can be improved by using other types of association rules), we have an idea about algorithm which computes, or updates, the matrix in the process of searching for factors instead of computing it once before the search. Second, we will look for a way how to use in the utilization of association matrix the so-called *essential elements* of the input data matrix, which are crucial for the GreEss algorithm [4] (which improves the GreConD algorithm).

References

[1] Agrawal R., Imieliński T., Swami A.: Mining association rules between sets of items in large databases, Proc. ACM SIGMOD 1993, 207–216.

[2] Bache K., Lichman M., UCI Machine Learning Repository [http://archive.ics.uci.edu/ml], Irvine, CA: University of California, School of Information and Computer Science, 2013.

[3] Belohlavek R., Vychodil V., Discovery of optimal factors in binary data via a novel method of matrix decomposition, J. Comput. Syst. Sci. 76(1)(2010), 3–20 (preliminary version in Proc. SCIS & ISCIS 2006).

[4] Belohlavek R., Trnecka M., From-below approximations in Boolean matrix factorization: Geometry and new algorithm, J. Comput. Syst. Sci. 81(8)(2015), 1678–1697.

[5] Ganter B., Wille R., Formal Concept Analysis: Mathematical Foundations, Springer, Berlin, 1999.

[6] Geerts F., Goethals B., Mielikäinen T., Tiling databases, Proc. Discovery Science 2004, 278–289.

[7] Hájek P., Holeňa M., Rauch J.: The GUHA method and its meaning for data mining. Journal of Computer and System Science 76(1)(2010), 34-–48.

[8] Hájek P., Havel I., Chytil M.: The GUHA method of automatic hypotheses determination. Computing 1(1966), 293—308.

[9] Hájek P., Havránek T.: Mechanizing Hypothesis Formation (Mathematical Foundations for a General Theory), Springer-Verlag 1978, 396 pp. New edition available online at `http://www.cs.cas.cz/~hajek/guhabook/`.

[10] Lucchese C., Orlando S., Perego R., Mining top-K patterns from binary datasets in presence of noise, SIAM DM 2010, 165–176.

[11] Miettinen P., Mielikäinen T., Gionis A., Das G., Mannila H., The discrete basis problem, IEEE Trans. Knowledge and Data Eng. 20(10)(2008), 1348–1362 (preliminary version in Proc. PKDD 2006).

[12] Myllykangas S. et al, 2006, DNA copy number amplification profiling of human neoplasms, Oncogene 25(55)(2006), 7324–7332.

[13] Rauch J.: Observational Calculi and Association Rules. Springer-Verlag, 2013.

[14] Rauch J., Šimůnek M.: Mining for 4ft rules, in: Proceedings of Discovery Science, Springer-Verlag, 2000.

[15] Tan P.-N., Steinbach M., Kumar V.: Introduction to Data Mining. Addison Wesley, Boston, MA, 2006.

[16] Xiang Y., Jin R., Fuhry D., Dragan F. F., Summarizing transactional databases with overlapped hyperrectangles, Data Mining and Knowledge Discovery 23(2011), 215–251 (preliminary version in Proc. ACM KDD 2008).