2016 IEEE 8th International Conference on Intelligent Systems

How to assess quality of BMF algorithms?

Radim Belohlavek
Department of Computer Science
Palacký University Olomouc
radim.belohlavek@acm.org

Jan Outrata
Department of Computer Science
Palacký University Olomouc
jan.outrata@upol.cz

Martin Trnecka
Department of Computer Science
Palacký University Olomouc
martin.trnecka@gmail.com

***Abstract*—We critically examine the problem of quality assessment of algorithms for Boolean matrix factorization. We argue that little attention is paid to this problem in the literature. We view this problem as a multifaceted one and identify key aspects with respect to which the quality of algorithms should be assessed. Because of its utmost importance, we focus on assessment of quality of sets of factors extracted from Boolean data, propose ways to assess such quality and provide experimental evaluation involving selected factorization algorithms. We argue that the views involved in our proposal, represent reasonable basic standpoints for further systematic approaches to quality assessment.**

***Keywords*—Boolean matrix; factorization; algorithm**

I. Introduction

A. Problem Setting

Boolean matrix factorization (BMF), called also Boolean matrix decomposition, has resulted in various methods for analysis and processing of Boolean data and has improved our understanding of this kind of data. The methods developed are becoming established tools for data management. Most research has focused on the design of new factorization strategies. Evaluation of performance of the developed algorithms has remained on intuitive grounds and has not been paid proper attention so far. It is the primary aim of this paper to look at evaluation of performance of BMF algorithms in detail.

Let us recall basic notions and introduce the notation we use regarding BMF. Denote by I an $n \times m$ Boolean matrix. The set of all $n \times m$ Boolean matrices is denoted $\{0,1\}^{n\times m}$. We interpret such matrices primarily as object-attribute incidence matrices (hence the symbol I). That is, the entry I_{ij} corresponding to the row i and the column j is either 1 or 0, indicating that the object i does or does not have the attribute j. The ith row and jth column vector of I is denoted by $I_{i\_}$ and $I_{\_j}$, respectively.

Generally speaking, 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

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

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

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

Importantly, a decomposition of I into $A \circ B$ may be interpreted as a discovery of k factors exactly or approximately explaining the data. The factor model given by (1) is described as follows: I, A, and B are interpreted as the object-attribute, object-factor, and factor-attribute matrices; the matrices A and B explain the object-attribute matrix I as follows: the object i has the attribute j 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 least k for which an exact decomposition $I = A \circ B$ exists is called the *Boolean rank* (or Schein rank) of I.

The approximate equality $\approx$ in (1) is assessed by means of the well-known L_1-norm $||\cdot||$,

$$||C|| = \textstyle\sum_{i,j=1}^{m,n} |C_{ij}|,$$

and the corresponding metric E, defined for $C, D \in \{0,1\}^{n\times m}$ by

$$E(C,D) = ||C - D|| = \textstyle\sum_{i,j=1}^{m,n} |C_{ij} - D_{ij}|. \qquad (2)$$

B. Relevant work

Matrix decompositions represent an extensive subject whose coverage is beyond the scope of this paper; we therefore focus on directly relevant work regarding Boolean matrices. Let us just mention that decompositions of Boolean matrices using methods designed originally for real-valued data and various modifications of these methods appear in a number of papers. [25] compares several approaches to assessment of dimensionality of Boolean data and concludes that a major problem with applying to Boolean data the methods designed originally for real-valued data is the lack of interpretability; similar observations were presented by other authors. Note also that in addition to the literature on Boolean matrices, results relevant to BMF are traditionally presented in the literature on binary relations, graph theory, and formal concept analysis, see e.g. [5], [8], [11], [23].

Among the first works on data analysis applications of BMF are [20], [21], in which the authors have already been aware of the provable computational difficulty (NP-hardness) of the decomposition problem due to the NP-hardness of the set basis problem [24]. An early algorithm, currently little known though, is the 8M algorithm, which is part of the BMDP statistical software since the late 1970s. In current data mining research, the interest in BMF is due to the work of Miettinen et al. In particular, the DBP, the corresponding complexity results, and the Asso algorithm discussed below

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

appeared in [16]. In [2], we showed that formal concepts (i.e. fixpoints of Galois connections) are natural factors of Boolean matrices, proved their optimality for exact factorizations, described transformations between attribute and factor spaces, and proposed the GRECOND algorithm discussed below. [4] presents a deeper insight into from-below approximations and a new algorithm based on it, as well as some observations regarding general BMF which we use. The other BMF algorithms which appeared in the recent data mining literature and which we use in our evaluation include HYPER [27] and PANDA [13]. Further work on various aspects of the decomposition problem and applications of BMF includes [7], [12], [14], [15], [17], [18], [22], [26]. The above literature contains numerous experimental evaluations of the proposed algorithms. In the evaluation in this paper we use common real benchmark data.

II. QUALITY ASSESSMENT

A. Variety of Aspects Regarding Quality Assessment

1) General Considerations: There are several aspects with respect to which one may address the question of quality of of factorization algorithms. They include the following ones:

- *computational complexity*, i.e. *time complexity* and *space complexity* of the algorithm in particular;
- *approximation factor* and possibly other characteristics regarding the capability of the algorithm to compute optimal and suboptimal solutions;
- *quality of factors* which itself is a complex aspect.

2) Computational Complexity and Approximation Factors: Computational complexity is a basic technical feature of a given factorization algorithm. *Ceteris paribus*, one clearly prefers algorithms with smaller complexity, i.e. those which require less time and space. It is a usual practice that the authors of new algorithms provide information about the time and possibly also space complexity, e.g. in terms of the standard big O notation. Since the estimates used are often loose and since the big O notation itself hides several issues, it may actually be more telling to provide information regarding relative time complexity, e.g. saying that algorithm 1 is on average (over a collection of certain datasets) three-times faster than algorithm 2, as recently done in [4]. Nevertheless, it appears that for most of the current factorization algorithms, neither time nor space complexity is a critical issue. That is to say, given the current ways of utilizing the results of BMF, the algorithms deliver the decompositions quickly enough—time and space complexity is therefore not prohibitive. Factorizations of benchmark datasets are being computed within seconds or small tens of seconds as a rule on an ordinary PC (see e.g. [4]). An exception is represented by the TILING algorithm [9] and the conceptually similar Algorithm 1 of [2] which both compute in advance a large space of possible factors and iteratively select factors in this set which is both time and space demanding.

Due to [24], it is known that the optimization version of the basic decomposition problem is NP-hard. The basic decomposition problem is to find for an input matrix $I \in \{0,1\}^{n\times m}$ two matrices $A \in \{0,1\}^{n\times k}$ and $B \in \{0,1\}^{k\times m}$ with $I = A \circ B$ such that k is as small as possible. NP-hardness applies to several variants of this problem as well, including the DBP and AFP mentioned below. As a result, unless P=NP, no polynomial time factorization algorithm computing optimal decompositions exists. Therefore, the existing algorithms are based on heuristics and one is interested in approximation factors. An *approximation factor* of an algorithm for an optimization problem represents a guarantee of suboptimality of the solution obtained by the algorithm [10]. For instance, if the approximation factor is 2 we are sure that the algorithm does compute at most $2k$ factors where k is the least number of factors possible (i.e. the Boolean rank) for the given input matrix. Generally, except for [9] and [2], there are almost no results on approximation factors of the proposed factorization algorithms in the literature. This is partially justified by the recent negative result on approximability [6] saying that the basic decomposition problem is NP-hard to approximate within factor $n^{1-\varepsilon}$. The lower approximability bound is therefore not encouraging; namely, note how bad a linear approximation factor, n, is for a potential algorithm: such factor only guarantees that, for example, when decomposing a 1000×1000 matrix with Boolean rank 50 (in which case $n = 1000$), the algorithm computes a decomposition with no more than $n \times 50 = 50\,000$ factors. The existing algorithms capable of computing exact decompositions, such as GRECOND, perform much better on benchmark datasets. From this viewpoint, the practical significance of obtaining the approximation factors is limited. On the other hand, clearly, analyses leading to approximation factors of the existing algorithms may clearly reveal substantial knowledge regarding the factorization problems and are thus needed.

We now turn to the third aspect, namely quality of factors. This aspect represents the main objective of our paper.

B. Quality of Factors

As was mentioned above, quality of factors represents a complex and important issue. The very question of whether a particular set of factors delivered by a particular algorithm for a particular dataset is good or not depends on circumstances. We first present a geometric view of factorization. Then we discuss the problem of interpretability of individual factors, in which we employ the geometric view, and the related knowledge discovery view of the problem of assessing the quality of factors. Then we turn to two other views, the reduction of dimensionality view and the explanatory view.

1) Geometry of Factorizations: It is a useful fact, made explicit e.g. in [4, Observation 1], that a decomposition of a Boolean matrix $I \in \{0,1\}^{n\times m}$ corresponds to a coverage of the entries of I containing 1s by rectangular matrices, or *rectangles* for short. These are matrices $J \in \{0,1\}^{n\times m}$, for which there exist vectors $C \in \{0,1\}^{n\times 1}$ and $D \in \{0,1\}^{1\times m}$ such that $J = C \circ D$, or alternatively, matrices whose entries with 1s form a rectangular area upon a suitable permutation of rows and columns. A Boolean matrix product $A \circ B$ with $A \in$

$\{0,1\}^{n\times k}$ and $B \in \{0,1\}^{k\times m}$ may alternatively be looked at as a max-superposition of rectangles $J_l = A_{\_l} \circ B_{l\_}$ in that

$$(A \circ B)_{ij} = \max_{l=1}^{k} (J_l)_{ij};$$

here, $A_{\_l}$ and $B_{l\_}$ denote the lth column of A and the lth row of B, respectively. This geometric view, which is unfortunately not always recognized in the literature on BMF, tells us that finding a decomposition (exact or approximate) of I into $A \circ B$ using k factors means finding a coverage (exact or approximate) of the 1s in I by k rectangles full of 1s.

2) Interpretability of Individual Factors: The view also tells us that no matter how one computes factors, *each factor*, say factor l, *may always be identified with a rectangle* J_l full of 1s, or alternatively, as a pair $\langle A_{\_l}, B_{l\_}\rangle$ of the abovementioned vectors, for which $J_l = A_{\_l} \circ B_{l\_}$. Such a factor is therefore naturally interpreted as an abstract property (or attribute), generally distinct from the m original attributes, which applies to some of the n objects, namely to objects i for which $A_{il} = 1$, and which is characterized by some of the m original attributes, namely attributes j for which $B_{lj} = 1$. Such attributes j are viewed as particular manifestations of factor l. This matter is described in detail in [2].

The above considerations imply that factors in BMF are easily interpretable. This is confirmed by several studies including [25] in which the authors argue that because of interpretability, BMF is considerably more appropriate to use than the many existing factorization methods originally developed for real-valued matrices. Interpretability is crucial for what may be called a *knowledge discovery view* of assessing quality of BMF algorithms—revealing easily interpretable factors from the input data means discovering new knowledge from the data.

There is, however, an important aspect regarding interpretability of factors in BMF, which is not properly understood in the literature. Namely, some authors such as those of the PaNDa algorithm [13], inspired by the well-known minimum description length principle (MDL), suggest that good factors are those with small description length. In our notation, a description length of factor $\langle A_{\_l}, B_{l\_}\rangle$ is the sum of the height and width of the corresponding rectangle, i.e. the number

$$dl(A_{\_l}, B_{l\_}) = ||A_{\_l}|| + ||B_{l\_}||,$$

which is the sum of the number of 1s in $A_{\_l}$ and the number of 1s in $B_{l\_}$. Minimizing the sum of description lengths of all the factors is in fact employed in the cost function of PaNDa. We claim that this view is flawed and that, instead, well interpretable factors should correspond to *maximal rectangles* rather than to rectangles that have small perimeter (i.e. small description length). The rationale for our argument stems *formal concept analysis* [8]. Put succinctly, it is generally not a good idea to remove attributes or objects from factors to shorten their description length because the interpretability of factors suffers. Our preference of maximal rectangles, articulated already in [2], derives from our experience with analyses of many datasets. We do not claim that the MDL approach is wrong. But we do claim that it should be justified by concrete data analyses. Otherwise it remains a theoretical construct—an unverified hypothesis regarding usefulness of factors. This issue along with illustrative examples shall be discussed in detail in the full version of this paper.

3) Quality of a Set of Extracted Factors: We recognize two ways of assessing the quality of a set of factors produced by a particular algorithm, which we call the *reduction of dimensionality view* and the *explanatory view*. These views are based on two generally recognized virtues of factorization methods. These matters are discussed in the next sections.

C. Reduction of Dimensionality View

Discovery of a set of k factors in the $n \times m$ Boolean matrix I may be viewed as a discovery of k new Boolean variables, i.e. new attributes. In particular, if $I \approx A \circ B$, then A describes the n given objects in terms of k factors (new attributes) and B describes a relationship between the k factors and the m original attributes. Thus, the objects may either be described in the m-dimensional space of the original attributes or in the k-dimensional space of factors. If the number k of factors is smaller than the number m of the original attributes, then going from the attribute space to the factor space may be viewed as what is known as reduction of dimensionality.

Generally speaking, the usefulness of factorization methods derives from the possibility to process data in the less-dimensional factor space rather than the original attribute space. The original attributes represent the directly observable features. Good factors, on the other hand, should capture the fundamental aspects of the data. If this is so, processing the data in the factor space turns out to be more efficient than processing in the original attribute space. Put conversely, if processing of data in the factor space is considerably better than the processing in the original attribute space, we may regard the factors as good ones. In this respect, one may then compare the quality of sets of factors delivered by particular factorization algorithms.

Surprisingly, even though considerable effort has been devoted to development of further and further BMF algorithms, almost no attention is paid to the question of how these methods may further be utilized in processing Boolean data, e.g. in machine learning. For one, this is in sharp contrast to factorization methods for real-valued data which are being routinely employed in machine learning and other areas. In addition, the lack of studies on possible utilization of dimensionality reduction due to BMF implies a lack of the much needed feedback for quality assessment of the various BMF algorithms.

An exception is, nevertheless, represented by [22] in which the author employs the transformation formulas between the factor and attribute spaces proposed in [2] in the problem of classification of Boolean data. Put briefly, the idea is to first factorize the classified Boolean data using a BMF algorithm and then to do classification in the factor space rather than in the original attribute space. Interestingly, such an approach results in a significant increase of classification accuracy. In

[3], the authors asked the question of which factorization methods perform well in the described scenario. It turned out that there are significant differences between the various factorization methods employed. That is to say, with respect to this particular employment of BMF, some algorithms turn out as good one while some as not so good. We omit the details of [3] due to limited scope but also because our main point here is to point out the fact that studies on employment of BMF in machine learning and other areas may not only demonstrate usefulness of BMF in general but also provide a useful, very concrete feedback regarding quality of BMF algorithms. Without such feedback, research in BMF algorithms might easily become a speculative theorizing.

D. Explanatory View

The other aspect pertaining to factorization methods relates to what may be called an *explanatory view.* It is based on the fact that the sole knowledge of factors is a useful knowledge regarding the data provided the factors explain the data well, because only then the factors represent the "true factors behind the data." In this respect, two particular views are recognized, which are represented by the discrete basis problem (DBP) and the approximate factorization problem (AFP). We first turn to the underlying idea that good factors should *explain data well.*

1) Explanation of Data by Factors: Given the input matrix $I \in \{0,1\}^{n\times m}$ and the object-factor and factor-attribute matrices $A \in \{0,1\}^{n\times k}$ and $B \in \{0,1\}^{k\times m}$, a basic way to measure how well the data es explained by the k factors is to observe the distance $E(I, A\circ B)$ defined by (2). The following additional characteristics are useful and will be used in sequel.

First, the distance (error) function E may naturally be split into two components, 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 $A \circ B$:

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

Second, to asses quality of decompositions delivered by the algorithms, we employ the following function of $A \in \{0,1\}^{n\times l}$ and $B \in \{0,1\}^{l\times m}$ representing the *coverage quality* of the first l factors delivered by the particular algorithm:

$$c(l) = 1 - E(I, A\circ B)/||I||. \tag{3}$$

Similar functions are used in [2], [4], [9], [16]. We observe the values of c for $l = 0, \dots, k$, where k is the number of factors delivered by a particular algorithm. In BMF, one is interested in the E_u and E_o parts of E (basically because of their nonsymmetric roles), and thus also in the corresponding relative errors

$$e_u = E_u(I, A\circ B)/||I||, \quad e_o(l) = E_o(I, A\circ B)/||I||. \tag{4}$$

Clearly,

$$c = 1 - E/||I|| = 1 - (E_u + E_o)/||I|| = 1 - e_u - e_o.$$

The value of c represents the overall coverage of data by the particular number l of observed factors. Clearly, for $l = 0$ (no factors added, A and B are "empty") we have $c = 0$, $e_u = 1$, and $e_o = 0$.

2) DBP—Importance of the First Few Factors: One view of BMF is reflected by the following problem:

- *Discrete Basis Problem* (DBP, [16]):
  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 few (presumably most important) factors. In this perspective, the quality of factors obtained by a BMF algorithms may be assessed by observing the values of coverage c for small numbers l of factors (e.g. for $l = 2, 5, 10$).

3) AFP—Importance of Explaining a Large Portion of Data: The second view of BMF is reflected by the following problem:

- *Approximate Factorization Problem* (AFP, [2]):
  Given I 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. In this perspective, the quality of factors obtained by a BMF algorithms may be assessed by observing the numbers l of factors needed to attain a prescribed coverage c (e.g. for $c = 0.8, 0.9, 1.0$).

4) Combined View—Toward Quality Metrics: The DBP and the AFP view represent two in a sense boundary views. Note that AFP is relevant when one desires a good representation of data by factors. This is indeed the case in the above-mentioned application of BMF as preprocessing method in classification of Boolean data. On the other hand, since Boolean factors are naturally interpreted as clusters (cf. the above section on interpretability of factors), the DBP view is relevant when a few informative clusters are sought in the data.

When evaluating quality of BMF algorithms, one is naturally led to a "combined view," reflecting both the DBP and the AFP views. Thus, a good BMF algorithm should produce a set of factors such that the first few factors have good coverage and, and the same time, the whole set of factors produced has a large coverage, possibly close to 1, i.e. the factors explain a large portion of data. Correspondingly, for a good factorization algorithm c as a function of the number l (the first l factors produced by the algorithm) should be increasing in l, should have relatively large values even for small l (i.e. should be steeply increasing in the beginning), and it is desirable that for $l = k$ we have $c(l)$ equal or close to 1, i.e. the data is almost fully explained by the k factors computed.

Our proposed measure of quality of a BMF algorithm is thus based on the coverage quality function $c(l)$ as defined by (3). However, in order to reflect the significance of DBP and AFP views, respectively, we introduce particular weights w_l

for the error function $E(I, A \circ B)$, and thus use a parameterized variant $c_w(l)$ of $c(l)$, which reads:

$$c_w(l) = 1 - w_l E(I, A \circ B)/||I||. \tag{5}$$

In order to emphasize the importance of the first factors in the DBP view, the weights reflecting this view should be increasing with increasing number l of factors. Therefore, the same amount of error is penalized more with each further factor. Analogously, in order to emphasize the need to explain a prescribed portion of data in the AFP view, the weights reflecting this view should be larger for a large value of error $E(I, A \circ B)$ and smaller for a small value of error. From the variety of many possible increasing functions we use simple linear functions for weights: $w_l = l/k$ for the DBP view and $w_l = 1 + (E(I, A \circ B) - \varepsilon)/(||I|| - \varepsilon)$ for the AFP view, where k are ε are the given values from the definitions of the DBP and AFP problems introduced above, respectively, and play a role of parameters here. In order to reflect the combined view, the weights are simply averaged: $w_l = (l/k + 1 + (E(I, A \circ B) - \varepsilon)/(||I|| - \varepsilon))/2$.

As noted above, the modified coverage $c_w(l)$ defined by (5) evaluates the quality of a decomposition consisting of the first l factors delivered by a particular BMF algorithm, not the quality of the algorithm alone. To evaluate the quality of a BMF algorithm for a particular input data, we need to evaluate the process of obtaining the final decomposition of input data by the algorithm, not the final decomposition only (to take into account that coverage is first steeply increasing and grows toward full coverage). Hence we need to take into account evaluations of all the decompositions produced by the algorithm during its run, from the decomposition consisting of no factors to the final decomposition consisting of l factors. As our measure of quality of a BMF algorithm for a given input data we propose the "area below the curve" of the function $c_w(j)$ ($j \in [0, l]$); see Figure 1 (for simplicity, all w_j's are considered 1 in the figure). Dividing the weights by the average weight $\sum_{j=0}^{l} w_j/(l+1)$ (to eliminate the cumulative effect of the weights) and scaling the area by the final number l of factors plus one we get the following variant of a *quality measure* of a BMF algorithm:

$$q = 1 - \left(\sum_{j=0}^{l} w_j \frac{E(I, A \circ B)}{||I||} \right) \Big/ \left(\sum_{j=0}^{l} w_j \right). \tag{6}$$

III. Experimental Evaluation

In this section, we provide experimental evaluation of quality of selected BMF algorithms on selected datasets. Due to limited scope, we restrict to the best known factorization algorithms and to selected real datasets only. Other algorithms and further datasets, including synthetic ones, shall be presented in a full version of this paper. For each algorithm, we observe its performance with respect to the DBP view, the AFP view, and the combined view as described above.

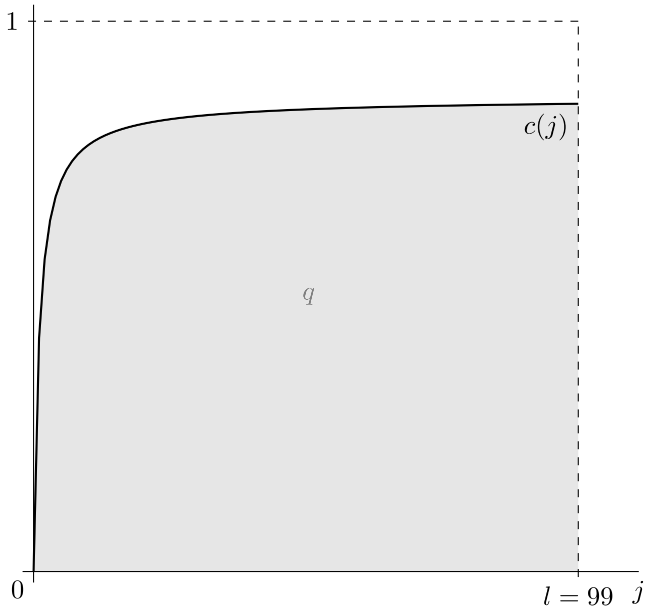

Fig. 1. Measure of quality of BMF algorithm

A. Datasets

We present results for the datasets Mushroom [1], Zoo [1], Paleo[1], DNA [19] and Firewall 1 [7], most of which are well known and used in the literature on BMF. The characteristics of the datasets shown in Table I are the number of objects × number of attributes (column Size), percentage of 1s in I (column Dens. 1), average value in the association matrix of I [16] (column Avg. assoc.) and the median of support of attributes (i.e. of the number of objects which have a given attribute; column Med. support).

TABLE I
Datasets and their characteristics

Dataset	Size	Dens. 1	Avg. assoc.	Med. support
DNA	4590×392	0.015	0.060	0.010
Firewall 1	365×709	0.124	0.459	0.154
Mushroom	8124×119	0.193	0.193	0.074
Paleo	501×139	0.051	0.092	0.042
Zoo	101×28	0.305	0.298	0.233

B. Algorithms

We now briefly describe the algorithms used in our comparison.

Asso [16], probably the most discussed BMF algorithm in the data mining literature. The algorithm is designed to solve the DBP and commits both types of errors, E_u and E_o.

GreConD [2, Algorithm 2] performs a particular greedy search "on demand" for formal concepts of the input matrix I which are used as factors. It is designed to compute exact decompositions. When stopped after computing the first k

[1] NOW public release 030717, available from http://www.helsinki.fi/science/now/.

factors or after the error E does not exceed ε, provides approximate solutions to both DBP and AFP.

NaiveCol [7] performs a greedy search for columns of the input matrix I. This is the simplest algorithm considered in this work. Like previous algorithm it can be used for both AFP and DBP.

Hyper [27] produces a set $\mathcal{F}$ of rectangles (called hyper-rectangles by the authors) in the input Boolean matrix I which provide an exact decomposition of I.

PaNDa (Patterns in Noisy Datasets) [13] is designed to solve a modification of DBP which consists in employing the MDLP.

GreEss [4] utilizes the formal concepts of I as factors again. However, it makes use of so-called essential part of I. It can be used for both AFP and DBP.

C. Assessment of Quality

Table II presents the numbers of factors required for prescribed values of the coverage function c defined by (3) and the coverage quality $c(k)$ achieved for prescribed numbers k of factors, representing the AFP and the DBP view of the decomposition quality evaluation, respectively. The prescribed values of c are given in %, i.e. $c = 80\%$ means $c = 0.8$. The entry "NA" for prescribed values of c means that the algorithm was not able to deliver a decomposition for the prescribed value. If the number of factors delivered was smaller than k, the coverage shown corresponds to the coverage of the set of factors delivered.

As we can see, PaNDa is not able to deliver decompositions for higher values of c. The algorithm produces decompositions of low coverage quality, in most cases smaller than 0.8, apparently because it is based on the MDL principle mentioned above instead of minimizing the error and thus maximizing coverage. Asso achieves high coverage quality for the first few factors but as the number of factors increases, its coverage is not so good compared to other algorithms, mainly due to the overcover error which cannot be eliminated once committed. Note also that GreEss algorithm delivers exact decompositions ($c = 100\%$) with the least number of factors.

The values of our proposed BMF algorithm quality measure q (6) for the evaluated algorithms and datasets are presented in Table III. In this table, q_c, for $c \in [0, 1]$, means that the function q was computed with the weights derived from the AFP view with the parameter ε corresponding to the coverage quality value c, i.e. $\varepsilon = (1-c) \cdot ||I||$. On the other hand, q_k, for $k > 1$, means that q was computed with the weights derived from the DBP view with parameter k. Finally, $q_{k,c}$ means q with both parameters, i.e. with k and ε corresponding to c as above, for the combined view of the quality evaluation.

We can see that GreConD algorithm ranked best both from the AFP and the DBP view, followed by GreEss algorithm with balanced values for the views. NaiveCol algorithm evaluates well in the AFP view but loses in the DBP view, while Asso algorithm performs the other way around – this is not surprising since the algorithm was designed for the DBP problem. PaNDa algorithm again evaluates very poorly.

TABLE II
Numbers of factors and coverage quality

Dataset		Asso	GreConD	NaiveCol	Hyper	PaNDa	GreEss
DNA	$c = 80\,\%$	75	106	144	181	NA	137
	$c = 90\,\%$	119	170	197	247	NA	190
	$c = 95\,\%$	173	241	242	302	NA	237
	$c = 100\,\%$	NA	511	368	391	NA	372
	$k = 10$	0.312	0.301	0.153	0.132	0.166	0.159
	$k = 20$	0.437	0.415	0.250	0.218	0.166	0.260
	$k = 30$	0.528	0.503	0.332	0.295	0.166	0.340
	$k = 40$	0.592	0.569	0.401	0.360	0.166	0.409
Firewall 1	$c = 80\,\%$	2	2	2	193	NA	2
	$c = 90\,\%$	3	4	4	223	NA	4
	$c = 95\,\%$	3	6	7	239	NA	6
	$c = 100\,\%$	NA	66	71	365	NA	64
	$k = 10$	0.917	0.981	0.976	0.083	0.491	0.981
	$k = 20$	0.922	0.992	0.991	0.151	0.491	0.992
	$k = 30$	0.924	0.996	0.996	0.203	0.491	0.997
	$k = 40$	0.925	0.998	0.998	0.254	0.491	0.998
Mushroom	$c = 80\,\%$	19	29	32	42	NA	31
	$c = 90\,\%$	34	46	47	57	NA	47
	$c = 95\,\%$	50	62	62	70	NA	61
	$c = 100\,\%$	NA	120	110	123	NA	105
	$k = 10$	0.556	0.582	0.512	0.285	0.346	0.546
	$k = 20$	0.652	0.715	0.674	0.502	0.346	0.696
	$k = 30$	0.720	0.812	0.789	0.664	0.346	0.793
	$k = 40$	0.765	0.873	0.862	0.780	0.346	0.865
Paleo	$c = 80\,\%$	83	86	83	83	NA	83
	$c = 90\,\%$	107	110	107	107	NA	106
	$c = 95\,\%$	122	127	122	122	NA	122
	$c = 100\,\%$	NA	151	139	139	NA	145
	$k = 10$	0.182	0.181	0.182	0.182	0.040	0.182
	$k = 20$	0.314	0.310	0.314	0.314	0.040	0.314
	$k = 30$	0.424	0.417	0.424	0.424	0.040	0.426
	$k = 40$	0.517	0.511	0.517	0.517	0.040	0.522
Zoo	$c = 80\,\%$	7	9	9	15	NA	9
	$c = 90\,\%$	10	13	13	19	NA	13
	$c = 95\,\%$	15	17	17	22	NA	16
	$c = 100\,\%$	NA	30	25	30	NA	25
	$k = 5$	0.694	0.703	0.603	0.412	0.524	0.660
	$k = 10$	0.868	0.853	0.834	0.652	0.539	0.849
	$k = 15$	0.922	0.927	0.937	0.824	0.539	0.943
	$k = 20$	0.943	0.973	0.985	0.928	0.539	0.985

IV. Conclusions

The aim of this paper is threefold. First, to point out an important problem in BMF, namely assessment of quality of BMF algorithms. Second, to identify key aspects of such assessment. Third, to propose quantitative ways to assess quality of BMF algorithms.

Our study revealed that quality assessment is paid a proper attention in the literature. In particular, one reason is a surprising lack of work in applications of BMF in machine learning, in spite of evidence of its usefulness, which contrasts with considerable amount of existing work in development of new BMF algorithms. We identified basic standpoints from which the assessment of quality may be approached and proposed three quantitative ways to assess quality. Two of them correspond to

TABLE III
BMF algorithm quality

Dataset		Asso	GreConD	NaiveCol	Hyper	PaNDa	GreEss
DNA	$q_{0.8}$	0.690	0.724	0.664	0.639	0.166	0.669
	$q_{0.9}$	0.756	0.782	0.712	0.678	0.166	0.719
	$q_{0.95}$	0.785	0.804	0.732	0.691	0.166	0.739
	q_1	0.797	0.852	0.745	0.699	0.166	0.753
	q_{10}	0.312	0.301	0.153	0.132	0.167	0.159
	q_{20}	0.437	0.414	0.250	0.218	0.167	0.260
	q_{30}	0.527	0.503	0.332	0.295	0.167	0.340
	q_{40}	0.591	0.568	0.400	0.359	0.167	0.408
	$q_{10,0.9}$	0.823	0.883	0.792	0.747	0.166	0.799
	$q_{20,0.8}$	0.821	0.880	0.787	0.741	0.166	0.794
Firewall 1	$q_{0.8}$	0.817	0.804	0.804	0.565	0.490	0.801
	$q_{0.9}$	0.886	0.909	0.901	0.600	0.490	0.907
	$q_{0.95}$	0.886	0.951	0.955	0.615	0.490	0.953
	q_1	0.923	0.996	0.996	0.627	0.490	0.996
	q_{10}	0.917	0.981	0.976	0.083	0.491	0.981
	q_{20}	0.922	0.992	0.991	0.150	0.491	0.992
	q_{30}	0.924	0.996	0.996	0.202	0.491	0.997
	q_{40}	0.925	0.998	0.998	0.253	0.491	0.998
	$q_{10,0.9}$	0.924	0.997	0.997	0.686	0.490	0.997
	$q_{20,0.8}$	0.924	0.997	0.997	0.678	0.490	0.997
Mushroom	$q_{0.8}$	0.622	0.740	0.729	0.657	0.344	0.733
	$q_{0.9}$	0.695	0.801	0.786	0.709	0.344	0.794
	$q_{0.95}$	0.725	0.827	0.810	0.728	0.344	0.819
	q_1	0.745	0.844	0.827	0.749	0.344	0.835
	q_{10}	0.556	0.582	0.511	0.285	0.346	0.545
	q_{20}	0.650	0.712	0.671	0.498	0.346	0.693
	q_{30}	0.715	0.805	0.781	0.654	0.346	0.786
	q_{40}	0.756	0.861	0.848	0.760	0.346	0.851
	$q_{10,0.9}$	0.764	0.876	0.863	0.798	0.344	0.870
	$q_{20,0.8}$	0.763	0.874	0.860	0.792	0.344	0.867
Paleo	$q_{0.8}$	0.567	0.565	0.567	0.567	0.040	0.570
	$q_{0.9}$	0.596	0.591	0.596	0.596	0.040	0.598
	$q_{0.95}$	0.605	0.600	0.605	0.605	0.040	0.607
	q_1	0.611	0.628	0.611	0.611	0.040	0.625
	q_{10}	0.181	0.181	0.181	0.181	0.040	0.181
	q_{20}	0.312	0.308	0.312	0.312	0.040	0.312
	q_{30}	0.419	0.412	0.419	0.419	0.040	0.420
	q_{40}	0.505	0.500	0.505	0.505	0.040	0.510
	$q_{10,0.9}$	0.662	0.679	0.662	0.662	0.040	0.677
	$q_{20,0.8}$	0.655	0.673	0.656	0.656	0.040	0.671
Zoo	$q_{0.8}$	0.723	0.741	0.699	0.628	0.515	0.731
	$q_{0.9}$	0.777	0.788	0.757	0.660	0.515	0.784
	$q_{0.95}$	0.805	0.808	0.779	0.671	0.516	0.802
	q_1	0.815	0.831	0.791	0.696	0.516	0.816
	q_5	0.690	0.699	0.599	0.409	0.522	0.656
	q_{10}	0.853	0.838	0.815	0.632	0.537	0.833
	q_{15}	0.898	0.899	0.899	0.772	0.537	0.910
	q_{20}	0.911	0.928	0.929	0.838	0.537	0.936
	$q_{10,0.9}$	0.844	0.864	0.832	0.745	0.521	0.853
	$q_{20,0.8}$	0.841	0.862	0.828	0.740	0.521	0.850

views known in the literature, the DBP and the AFP view. The third one is new and corresponds to a combined view which represents a natural requirement for a good BMF algorithm. Our experimental evaluation demonstrates that the proposed ways to assess quality are reasonable and demonstrates how some of the best-known algorithms fare with respect to the three views.

Further research needs to include the following topics: development of further quantitative ways to quality assessment; employment of BMF in machine learning and other fields to obtain feedback regarding quality of BMF algorithms from concrete employment of factors; development of further criteria for quality assessment, such as the often discussed capability of BMF algorithms to deal with noise in the input data.

References

[1] 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.

[2] 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).

[3] Belohlavek R., Outrata J., Trnecka M., Impact of Boolean factorization as preprocessing methods for classification of Boolean data, Ann. Math. Artif. Intell. 72(1–2)(2014), 3–22.

[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] Brualdi R. A., Ryser H. J., Combinatorial Matrix Theory, Cambridge University Press, 1991.

[6] Chalermsook P., Heydrich S., Holm E., Karrenbauer A.: Nearly tight approximability results for minimum biclique cover and partition. ESA 2014, pp. 235–246.

[7] Ene A. et al., Fast exact and heuristic methods for role minimization problems. Proc. SACMAT 2008, pp. 1–10.

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

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

[10] Hromkovic J. *Algorithmics for Hard Problems. Introduction to Combinatorial Optimization, Randomization, Approximation, and Heuristics.* Springer, Berlin, 2010.

[11] Kim K.H., Boolean Matrix Theory and Applications, M. Dekker, NY, 1982.

[12] Lu H., Vaidya J., Atluri V., Hong Y., Constraint-aware role mining via extended Boolean matrix decomposition, IEEE Trans. Dependable and Secure Comp. 9(5)(2012), 655–669.

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

[14] Miettinen P., The Boolean column and column-row matrix decompositions, Data Mining and Knowledge Discovery 17(2008), 39–56.

[15] Miettinen P., Sparse Boolean matrix factorizations, Proc. IEEE ICDM 2010, pp. 935–940.

[16] 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).

[17] Miettinen P., Vreeken J., Model order selection for Boolean matrix factorization, Proc. ACM SIGKDD 2011, pp. 51–59.

[18] Monson S. D., Pullman S., Rees R., A survey of clique and biclique coverings and factorizations of (0,1)-matrices, Bull. ICA 14(1995), 17–86.

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

[20] Nau D.S., Specificity covering, Tech. Rep. CS-1976-7, Duke University, 1976.

[21] Nau D.S., Markowsky G., Woodbury M.A., Amos D.B., A mathematical analysis of human leukocyte antigen serology, Math. Biosci. 40(1978), 243–270.

[22] Outrata J., Boolean factor analysis for data preprocessing in machine learning, Proc. ICMLA 2010, pp. 899–902.

[23] Schmidt G., Relational Mathematics, Cambridge University Press, 2011.

[24] Stockmeyer L., The set basis problem is NP-complete, Tech. Rep. RC5431, IBM, Yorktown Heights, NY, USA, 1975.

[25] Tatti N., Mielikäinen T., Gionis A., Mannila H., What is the dimension of your binary data?, Proc. IEEE ICDM 2006, pp. 603–612.

[26] Vaidya J., Atluri V., Guo Q., The role mining problem: finding a minimal descriptive set of roles, Proc. SACMAT 2007, pp. 175–184.

[27] 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).