A COMPARATIVE STUDY OF TWO METHODOLOGIES FOR BINARY DATASETS ANALYSIS

Alexander Frolov[*], *Dušan Húsek*[†], *Pavel Y. Polyakov*[‡], *Hana Řezanková*[§]

Abstract: Studied are differences of two approaches targeted to reveal latent variables in binary data. These approaches assume that the observed high dimensional data are driven by a small number of hidden binary sources combined due to Boolean superposition. The first approach is the Boolean matrix factorization (BMF) and the second one is the Boolean factor analysis (BFA). The two BMF methods are used for comparison. First is the M8 method from the BMDP statistical software package and the second one is the method suggested by Belohlavek & Vychodil. These two are compared to BFA, especially with the Expectation-maximization Boolean Factor Analysis we had developed earlier has, however, been extended with a binarization step developed here. The well-known bars problem and the mushroom dataset are used for revealing the methods' peculiarities. In particular, the reconstruction ability of the computed factors and the information gain as the measure of dimension reduction was under scrutiny. It was shown that BFA slightly loses to BMF in performance when noise-free signals are analyzed. Conversely, BMF loses considerably to BFA when input signals are noisy.

Key words: *Dimension reduction, statistics, data mining, Boolean factor analysis, Boolean matrix factorization, information gain, likelihood-maximization, bars problem*

Received: November 19, 2012
Revised and accepted: December 10, 2012

[*] Alexander Frolov
Institute of Higher Nervous Activity and Neurophysiology, RAS, Butlerova 5a, Moscow, Russia, E-mail: aafrolov@mail.ru

[†] Dušan Húsek
Institute of Computer Science, Academy of Sciences of the Czech Republic, Pod Vodárenskou věží 2, Prague 8, Czech Republic, E-mail: dusan@cs.cas.cz

[‡] Pavel Polyakov
Institute of Higher Nervous Activity and Neurophysiology, RAS, Butlerova 5a, Moscow, Russia, Department of Computer Science, VŠB-Technical University of Ostrava, 17. listopadu 15, Ostrava – Poruba, Czech Republic, E-mail: pavel.polyakov@vsb.cz

[§] Hana Řezanková
Department of Statistics and Probability, University of Economics, W. Churchill Sq. 4, 130 67 Prague 3, Czech Republic, E-mail: rezanka@vse.cz

1. Introduction

A fundamental problem in many data-analysis tasks is to find a suitable representation of the data. A useful representation typically makes the latent structure in the data explicit, and often reduces the dimensionality of the data so that further computational methods can be applied. Dimensionality reduction methods are able to transform a high-dimensional space of attributes to a lower-dimensional space. There exists high demand for such transformation in many areas of human activity (such as engineering, computer science, biology or economics) where one is facing a problem of efficient processing of large datasets. So dimension reduction methods are a crucial part of our life.

The most prevalent model of the latent data structure is the latent factor model which explains the dependencies in high-dimensional data by positing the existence of hidden variables representing common causes (factors) for observed variables. Two approaches are developed to reveal such a latent structure of the data: factorization and factor analysis. Methods of factorization are applied to a given dataset represented in the form of $M \times N$ matrix $\mathbf{X}$ containing M N-dimensional observations. Data factorization aims at decomposition of $\mathbf{X}$ into $M \times L$ matrix $\mathbf{S}$ and $L \times N$ matrix $\mathbf{F}$ with $L < N$ such that $\mathbf{X}$ could be approximately represented as $\mathbf{SF}$. Then one can say that $\mathbf{X}$ is decomposed into L factors whose relation to the observable variables is given by rows of matrix $\mathbf{F}$ as factor loadings and rows of matrix $\mathbf{S}$ as factor scores indicate how strongly each factor is related to each observation. Singular Value Decomposition (SVD) [12] or Nonlinear Matrix Factorization (NMF) [14, 18] are typical examples of such a decomposition. The methods of factorization aim to find $\mathbf{S}$ and $\mathbf{F}$ that minimize the deviation $\epsilon = ||\mathbf{X} - \mathbf{SF}||$ for a given number of factors $L < \min(M, N)$ or to find $\mathbf{S}$ and $\mathbf{F}$ that provide a given deviation ϵ with the smallest number $L < \min(M, N)$.

Methods of factor analysis deal with the same matrix of observations $\mathbf{X}$ but do not imply the best approximation by $\mathbf{SF}$, on the contrary they imply that $\mathbf{X}$ is generated according to some generative model so that each observation consists of regular and noisy parts. Then the aim of factor analysis is to find the regular parts of factors $\mathbf{F}$, to find the matrix $\mathbf{S}$, entries which indicate how strongly each factor is related to each observation, and to find statistical parameters of noise. Assuming that some entries of $\mathbf{X}$ are distorted by noise, neither the closest $\mathbf{X}$ approximation nor the best its compression is desirable. The goal is to reveal the structure underlying the data. Since the structure is regular and noise is irregular, factor analysis provides not only structure recovery in observations of a given data set but also its recovery in new observations if they correspond to the same generative model. Thus, factorization methods target to solve the “lossy compression problem” and methods of factor analysis to solve “structure inference problem” [7]. The classical examples of factor analysis are methods based on the normal linear generative model [13, 3].

If the input matrix $\mathbf{X}$ is binary, it is natural to require that $\mathbf{S}$ and $\mathbf{F}$ are also binary. In this paper, we consider the methods satisfying this requirement. In this case, the combination operation of matrices $\mathbf{S}$ and $\mathbf{F}$ is the Boolean matrix product (i.e., the matrix product in the semiring of Boolean $\bigwedge$ and $\bigvee$). The typical examples of Boolean factorization are the methods developed in [16, 17, 4]. The Boolean factor analysis is exemplified by Noisy-OR Component Analysis (NOCA) [19], Multi-Assignment Clustering (MAC) [7] and Expectation-Maximization Boolean Factor Analysis (EMBFA) [10]. To compare these two approaches we used two methods of Boolean factorization developed in [16, 4] and the EMBFA method [10] of Boolean factor analysis.

The paper is organized as follows. In Section 2, we briefly describe the three methods chosen for comparison. Section 3 contains a description of the two measures used for estimating the methods' performance. Then Section 4 contains a description of the databases used. Next, an experimental comparison of the methods is presented in Section 5. Finally, Section 6 contains discussion on further issues and concludes the paper.

2. BMF and BFA Methods

Boolean matrix factorization [4] implies presentation of binary matrix of observed dataset $\mathbf{X}$ in the form

$$\mathbf{X} = \mathbf{S} \otimes \mathbf{F}, \tag{1}$$

where each row of binary $M \times N$ matrix $\mathbf{X}$ is the observed pattern, each row of binary $L \times N$ matrix $\mathbf{F}$ is the representation of a factor in the signal space and each row of binary $M \times L$ matrix $\mathbf{S}$ is the set of factor scores defining which factors are mixed in the patterns. Boolean matrix product $\otimes$ means that each component of matrix $\mathbf{X}$ is obtained as $x_{mj} = \bigvee_{i=1}^{L} s_{mi} f_{ij}$. The method implies identification of a minimal set of factors that provide representation of the observed data in the form (1). The number of such factors is called the Boolean rank of $\mathbf{X}$. Since the combinatorial problem of $\mathbf{X}$ rank identification is NP complete [17] the existing methods give reasonable, but not obligatory optimal solutions. The optimal solution is given by a brute force search which is not suitable for high dimensional data. On the other side, the classical linear methods could not take into account non-linearity of Boolean summation and are, therefore, inadequate for this task.

2.1 Formal concepts Boolean matrix factorization

Recently, authors Belohlavek and Vychodil revealed in [4] a tight relationship between BMF and formal concept analysis [11], and developed two simple greedy algorithms solving this task.

2.1.1 The BVA1 algorithm

The BVA1 algorithm, named as "Algorithm 1" in [4], utilizes formal concepts of $\mathbf{X}$ as factors. Recall that a formal concept of $\mathbf{X}$ [11] is any pair $\langle C; D\rangle$ of sets $C \subseteq 1, \dots, M$ (rows, objects) and $D \subseteq 1, \dots, N$ (columns, attributes) satisfying the following property: D is the set of all attributes j for which $x_{mj} = 1$ for every object $m \in C$ and, vice versa, C is the set of all objects m for which $x_{mj} = 1$ for every attribute $j \in D$. Formal concepts are very well understood by domain experts. Geometrically, they are, up to permuting rows and columns, just maximal rectangles full of Ones in the matrix X. If a set $\Im$ of formal concepts is to be used as a set of factors of $\mathbf{X}$, the corresponding matrices $\mathbf{S}_\Im$ and $\mathbf{F}_\Im$ are defined in the following way: the ith column of $\mathbf{S}_\Im$ is just the characteristic vector of C_i and the ith row of $\mathbf{F}_\Im$ is just the characteristic vector of D_i, where $\langle C_i; D_i\rangle$ is the lth formal concept in $\Im$. It is proved in [11] that using such factors is optimal in that the Boolean rank of $\mathbf{X}$ may be achieved by using formal concepts as factors. In BVA1, one first computes all the formal concepts of $\mathbf{X}$. The algorithm proceeds in a greedy way: In every step, it selects the concept that covers the largest number of entries with One in $\mathbf{X}$ that were not covered by the previously selected concepts ($\langle C; D\rangle$ covers $x_{mj} = 1\rangle$ if $m \in C$ and $j \in D$, i.e. the rectangle corresponding to $\langle C; D\rangle$ spans over the entry $\langle m; j\rangle$).

2.1.2 The BVA2 algorithm

The BVA2 algorithm, named as “Algorithm 2” in [4], utilizes formal concepts of $\mathbf{X}$ as factors in the same way as algorithm BVA1. However, algorithm BVA2 avoids the necessity to compute all the concepts of $\mathbf{X}$ and browse through them during greedy selection. Instead, BVA2 computes the candidate factors, i.e. concepts of $\mathbf{X}$, on demand by the following greedy procedure. Each time a new factor is needed, one looks at the columns of $\mathbf{X}$ and selects the concept generated by a column which covers most of the yet uncovered Ones in $\mathbf{X}$. Such a concept corresponds to a narrow but high rectangle in the data. Then, one seeks whether such a rectangle may be extended to a wider rectangle by adding some attribute and deleting the objects so that one still has a rectangle. If so, one selects as the best such a rectangle, i.e. the rectangle covering most of the yet uncovered Ones in $\mathbf{X}$. One repeats the process of extension until no such extension yields a better rectangle. In this way one obtains the new factor and eventually a set $\mathbf{F}$ of formal concepts – the factors of $\mathbf{X}$.

For our computer comparison we used the second faster BVA2 algorithm only. In the limit of large M, N and L the number of its operations is proportional to

$$\Omega_1 = MLN^2\langle n_f\rangle\langle p_j\rangle, \tag{2}$$

where $\langle n_f\rangle$ is a mean number of Ones in factors and $\langle p_j\rangle$ is a mean probability for each component to be One in a data set. When using the programme implemented in C++ and PC Core2 6400, 2.13 GHz the execution time of one operation amounts to about 10^{-11} sec.

2.2 The M8 procedure of BMDP

The second approach to Boolean matrix factorization is implemented in the statistical package BMDP [1], which was originally developed at UCLA for biomedical applications. The M8 procedure of BMDP is sufficiently effective despite being based on the brute force search approach combined with the Boolean regression procedures.

To compute L factors of $\mathbf{X}$ (and thus the corresponding $M \times L$ and $L \times N$ matrices $\mathbf{S}$ and $\mathbf{F}$), the algorithm starts with $k < L$ candidate rows of $\mathbf{F}$ (candidate factor loadings). These are either supplied by the user or computed from $\mathbf{X}$ using a heuristic based on inclusion of the columns of $\mathbf{X}$. From this set of k vectors of factor loadings, the algorithm computes k vectors of factor scores (candidate columns of $\mathbf{S}$); from the k vectors of scores, the algorithm tries to find better k vectors of factor loadings, etc. until no change occurs or three such cycles are completed. Such tuning of factor loadings and scores is called refinement (the details are too technical to be included here). The algorithm then iteratively adds further factors as follows. Suppose l factors have been obtained. Then, one adds new factor $l+1$, refines the loadings and scores of all the factors as above, adds new factor $l+2$ and refines again. Then, the lth factor is removed and the remaining factors are refined. Consequently, the process is repeated, i.e. two new factors are added, one is removed, etc. For example, starting with $k = 2$ factors, we obtain 2, 3, 4, 3, 4, 5, 4, 5, 6, 7, 6, 7, 8, etc. factors. The process stops when the required number L of factors is obtained the second time. For example, with $k = 2$ and $L = 6$, one computes 2, 3, 4, 3, 4, 5, 4, 5, 6, 7, 6 factors and the last six ones are the final factors output by the algorithm. By default, the initial $k = L-2$ but the initial k may also be set by the user. A new factor is added based on the matrix describing the error committed by the factors obtained so far. In particular, one uses the column of $\mathbf{X}$ which contains the largest number of Ones uncovered by the previously computed factors. The execution time for this algorithm is approximately the same as for BVA2.

2.3 Expectation-maximization Boolean factor analysis

The Expectation-maximization method for Boolean factor analysis was developed for analysis of data of statistical origin [10]. Similar to BMF, in terms of BFA, each observation is a binary row vector $\mathbf{x} = [x_1, \ldots, x_N]$, each factor $\mathbf{f}_i = [f_{i1}, \ldots, f_{iN}]$ is a binary row vector of dimension N. The factor is supposed to have two or more nonzero components, therefore it is called a common factor [3]. Unlike BMF, BFA is a statistical method that assumes the existence of a probabilistic generative model. We suppose that the most appropriate for BFA and the most close to the classical linear factor analysis is the Noisy-OR generative model. Its parameters are $\boldsymbol{\Theta} = (p_{ij}, q_j, \pi_i, i = 1, \ldots, L, j = 1, \ldots, N)$. Parameter p_{ij} is the probability of the jth attribute presence in an observation due to the ith common factor. For attributes constituting the common factor, i.e., for attributes with $f_{ij} = 1$, the probability p_{ij} is high, and for the other attributes (with $f_{ij} = 0$), it is zero. Thus, the contribution of the ith common factor is defined by the binary row vector $\mathbf{f}_i' = [f_{i1}', \ldots, f_{iN}']$ which is a distorted version of the factor loadings vector $\mathbf{f}_i$. Factor distortion implies that the entries of $\mathbf{f}_i$ having the value equal to One can change their values to Zero with probability $1 - p_{ij}$ but none of the entries of $\mathbf{f}_i$ equal to Zero can change their value to One.

In addition to L common factors, the observations are supposed to be influenced by N specific factors, each having only one nonzero component η_j [3]. Parameter q_j is the probability of the presence of the jth attribute in an observation due to the jth specific factor. The total contribution of the specific factors into an observation is defined by a binary row vector $\boldsymbol{\eta} = [\eta_1, \ldots, \eta_N]$. Formally the introduction of specific factors is equivalent to the introduction of a special common factor $\mathbf{f}_0$ with all entries equal to One, appearing in all observations and having

$p_{0j} = q_j$. However, this special factor does not carry any information concerning the hidden regular structure of a given dataset and, thus, its introduction contradicts the designation of common factors in BFA. Moreover, to stress the distinction of common and specific factors it is required to put that number of Ones in each common factor is less than N.

As a result, any observation $\mathbf{x}$ can be presented in the form

$$x_j = \left[\bigvee_{i=1}^{L} s_i \wedge f'_{ij} \right] \vee \eta_j, \tag{3}$$

where $\mathbf{s} = [s_1, \ldots, s_L]$ is a binary row vector of factor scores of dimension L, L being the total number of common factors.

It is worth noting that the notions of common and specific factors are inherent for factor analysis but not for matrix factorization. Particularly, it is not specially required that each row of matrix $\mathbf{F}$ in presentation $\mathbf{X}$ according to (1) has more than one and less than N nonzero component. However, in spite of the differences both approaches provide very similar decompositions of a given dataset $\mathbf{X}$ in the form (1) or (3).

Parameter π_i $(i = 1, \ldots, L)$ is the probability that the ith factor appears in an observation.

We assume that common factors are distorted independently of other common factors and specific factors, common factor's components are distorted independently of other components, and specific factors are independent of each other and of the common factors.

The same generative model is used in NOCA [19] while MAC is based on Mixture Noise Model [7].

In contrast to BMF, which is aimed to find exact or approximate decomposition of a given dataset, the aim of BFA is to find the parameters of a generative model $\mathbf{\Theta}$ and factor scores for all patterns of the dataset. Moreover, it is supposed that the factors found could also be detected in any arbitrary pattern $\mathbf{x}$, if generated by the same BFA model. Note that in the case $p_{ij} = f_{ij}$ and $q_j = 0$ BFA provides the exact decomposition of a given dataset equivalent to BMF solution given by (1), however, as for BMF there is no guarantee that this decomposition is optimal.

As NOCA and MAC EMBFA maximizes the likelihood of the observed data by maximizing the free energy which in the case of described generative model has the form

$$\mathcal{F} = \sum_{m=1}^{M} \left\{ \sum_{\mathbf{s}} g_m(\mathbf{s}) \log(P(\mathbf{x}_m|\mathbf{s}, \mathbf{\Theta}) P(\mathbf{s}|\mathbf{\Theta})) + H(g_m(\mathbf{s})) \right\},$$

where $g_m(\mathbf{s})$ is the expected distribution of factor scores for the mth pattern, $H(g_m(\mathbf{s}))$ is the Shannon entropy of $g_m(\mathbf{s})$,

$$P(\mathbf{x}|\mathbf{s}, \mathbf{\Theta}) = \prod_{j=1}^{N} P(x_j|\mathbf{s}, \mathbf{\Theta}),$$

where

$$P(x_j|\mathbf{s},\mathbf{\Theta}) = x_j - (2x_j - 1)(1 - q_j)\prod_{i=1}^{L}(1 - p_{ij})^{s_i}, \tag{4}$$

$$P(\mathbf{s}|\mathbf{\Theta}) = \prod_{i=1,L} \pi_i^{s_i}(1 - \pi_i)^{1-s_i}.$$

The iterations of the Expectation-Maximization (EM) algorithm alternatively increase $\mathcal{F}$ with respect to the distributions g_m, while holding $\mathbf{\Theta}$ fixed (the E-step), or with respect to parameters of the model $\mathbf{\Theta}$, while holding g_m fixed (the M-step) [5].

At the E-step, the distributions g_m maximizing $\mathcal{F}$ are calculated according to the following equation

$$g_m(\mathbf{s}|\mathbf{\Theta}) = \frac{P(\mathbf{x}_m|\mathbf{s},\mathbf{\Theta})P(\mathbf{s}|\mathbf{\Theta})}{\sum_{\mathbf{s}} P(\mathbf{x}_m|\mathbf{s},\mathbf{\Theta})P(\mathbf{s}|\mathbf{\Theta})}.$$

The obtained distributions g_m provide the expected likelihood of the observed data over the factor scores for the given set of parameters of the generative model.

At the M-step, π_i can be obtained as

$$\pi_i = (1/M)\sum_{m=1}^{M} s_{mi},$$

where

$$s_{mi} = \sum_{\mathbf{s}} g_m(\mathbf{s}|\mathbf{\Theta})s_i.$$

Respectively, p_{ij} and q_j can be obtained by steepest ascent maximization of $\mathcal{F}$:

$$\Delta p_{ij} = \gamma_{ij}\frac{\partial \mathcal{F}}{\partial p_{ij}}, \qquad \Delta q_j = \gamma_j\frac{\partial \mathcal{F}}{\partial q_j}, \tag{5}$$

where γ_{ij} and γ_j are learning rates,

$$\begin{aligned}\frac{\partial \mathcal{F}}{\partial p_{ij}} &= \sum_{m=1}^{M}\sum_{\mathbf{s}} g_m(\mathbf{s}|\mathbf{\Theta})P(x_{mj}|\mathbf{s},\mathbf{\Theta})^{-1}\frac{\partial P(x_{mj}|\mathbf{s},\mathbf{\Theta})}{\partial p_{ij}}\\ \frac{\partial \mathcal{F}}{\partial q_j} &= \sum_{m=1}^{M}\sum_{\mathbf{s}} g_m(\mathbf{s}|\mathbf{\Theta})P(x_{mj}|\mathbf{s},\mathbf{\Theta})^{-1}\frac{\partial P(x_{mj}|\mathbf{s},\mathbf{\Theta})}{\partial q_j}\end{aligned}$$

and according to (4)

$$\begin{aligned}\frac{\partial P(x_{mj}|\mathbf{s}_m,\mathbf{\Theta})}{\partial p_{ij}} &= (x_{mj} - P(x_{mj}|\mathbf{s}_m,\mathbf{\Theta}))\frac{s_{mi}}{1 - p_{ij}}\\ \frac{\partial P(x_{mj}|\mathbf{s}_m,\mathbf{\Theta})}{\partial q_j} &= (x_{mj} - P(x_{mj}|\mathbf{s}_m,\mathbf{\Theta}))\frac{1}{1 - q_j}.\end{aligned}$$

At each iteration cycle of the M-step, we put $p_{ij} = 0$ if

$$p_{ij} < 1 - \prod_{l \neq i}(1 - \pi_l p_{lj}),$$

where the right side of the inequality is the probability that the jth attribute appears in the pattern due to other factors besides $\mathbf{f}_i$.

In our computer experiments, we set the learning rates in (5) to be

$$\gamma_{ij} = p_{ij}(1 - p_{ij})/(M\pi_i), \qquad \gamma_j = q_j(1 - q_j)/M.$$

We found empirically that they provide convergence of the steepest ascent procedure for few steps.

The steepest ascent procedure (5) at each M-step continues until $\sum_{ij} |\Delta p_{ij}|/LN$ becomes smaller than $\epsilon_2 = 10^{-3}$.

The obtained values of p_{ij}, q_j and π_i are used as the input for the next E-step. The EM iterative procedure terminates once values $\sum_{ij} |\Delta p_{ij}|/LN$ remained smaller than $\epsilon_1 = 10^{-3}$, where Δp_{ij} is the change of the model parameters p_{ij} comparing to the outcome of the previous M–step.

The described procedure provides a reasonable but not obligatory optimal solution. The global maximization of likelihood function is not so desired. Our experiments with artificial data have shown that likelihood function is slightly sensitive to some accidental peculiarities of a given dataset. Thus, its highest value does not obligatorily correspond to the most precise solution.

After the convergence of the procedure, the resulting values s_{mi} are the estimates of the factor scores which are not binary but gradual. To satisfy the generative model, we binarized those values. The binarization threshold was chosen to maximize the BFA information gain [8, 9] (see below).

As suggested in [15] for EM applied to maximal causes model, we restricted the EMBFA algorithm to the case of sparse scores, when only a small number of factors (no more than three) are supposed to be mixed in the observed patterns. In this case, summation over $\mathbf{s}$ in the above formulas is reduced to

$$\begin{aligned}\sum_{\mathbf{s}}(\dots) &= (\dots)_{\mathbf{s}=0} + \sum_i(\dots)_{\mathbf{s}=\mathbf{s}_i} + \sum_{i<j}(\dots)_{\mathbf{s}=\mathbf{s}_{ij}} + \\ &+ \sum_{i<j<k}(\dots)_{\mathbf{s}=\mathbf{s}_{ijk}},\end{aligned} \tag{6}$$

where $\mathbf{s}_i$ is the vector of factor scores with all zeros except s_i, $\mathbf{s}_{ij}$ is the vector of factor scores with all zeros except s_i and s_j, and $\mathbf{s}_{ijk}$ is the vector of factor scores with all zeros except s_i, s_j and s_k. An increase of the number of terms in (6) leads to a considerable rise in computational complexity. In the limit of large M, N and L the number of operations required for one iterative step is proportional to

$$\Omega_2 = MNL^3\langle p_j \rangle, \tag{7}$$

where $\langle p_j \rangle$ is a mean probability for each component to be One in a data set. When using the programme implemented in C++ and PC Core2 6400, 2.13 GHz the execution time of one operation amounts to about $5 \cdot 10^{-9}$ sec.

To start the EM procedure we set $\pi_i = 1/L$, where L is expected number of factors that have to be set in advance; we also initialized $q_j = 0$ and p_{ij} with random values uniformly distributed in the range from 0.2 to 0.7. Sometimes (in five percent of cases) the EM procedure converged for a few steps to a zero solution. This problem could presumably be overcome by a better choice of initial parameters. But we did not optimize their choice and simply restarted EM when it failed.

In terms of BMF, the matrix of factor scores $\mathbf{S}$, whose rows are vectors of factor scores $\mathbf{s}_i$ $(i = 1, \ldots, M)$, is the object-factor matrix $\mathbf{S}$. To estimate factor-attribute matrix $\mathbf{F}$, we binarized probabilities p_{ij} assuming that $f_{ij} = 1$ if $p_{ij} \geq p_{th}$ and $f_{ij} = 0$ if $p_{ij} < p_{th}$, where p_{th} is the binarization threshold. The binarization threshold was chosen to maximize coverage quality (see below).

3. Estimation of the Methods Performance

To compare the efficiency of the three methods we used two measures for estimating their performance: information gain and coverage quality. The first measure is based on statistics of the input data and relates to BFA while the second one relates to BMF.

3.1 Information gain

Information gain is a general information theoretic measure of BFA efficiency, which is a difference of two entropies. The first is the entropy of a dataset when its hidden factor structure is unknown, and the second is the entropy when it is revealed and taken into account [8].

If factor structure of the signal space is unknown, then representing the jth component of vector $\mathbf{x}$ requires $h(p_j)$ bits of information, where $h(x) = -x\log_2 x - (1-x)\log_2(1-x)$ is the Shannon function and p_j is the probability of the jth component to take One. Representing the whole dataset requires

$$H_0 = M\sum_{j=1}^{N} h(p_j)$$

bits of information. If the hidden factor structure of the signal space is detected (that is, all generative model parameters and all factor scores in the dataset are found), then representing the whole dataset requires

$$H = H_1 + H_2$$

bits of information. Here

$$H_1 = M\sum_{i=1}^{L} h(\pi_i)$$

defines information required to represent factor scores and

$$H_2 = \sum_{m=1}^{M}\sum_{j=1}^{N} h(P(x_{mj}|\mathbf{\Theta}, \mathbf{S}_m)) \tag{8}$$

defines information required to represent all patterns of the dataset when factor scores are given. In (8), $P(x_{mj}|\mathbf{\Theta}, \mathbf{S}_m)$ is the probability of the jth component of the mth signal $\mathbf{x}_m$ to take the value x_{mj}. This probability is given by (4).

We defined the relative information gain as

$$G = (H_0 - H)/H_0.$$

As shown in [8], information gain decreases when noise in the dataset increases and/or the errors in the BFA solution increases. Thus, it is a reliable measure of the BFA quality and suitability of BFA to a given dataset in principle.

3.2 Coverage quality

According to (1), the product $\mathbf{S} \otimes \mathbf{F}$ should approximately cover the input matrix $\mathbf{X}$. The error between $\mathbf{X}$ and its coverage by $\mathbf{S} \otimes \mathbf{F}$ is a natural measure of BMF quality. The error $E(\mathbf{X}, \mathbf{S} \otimes \mathbf{F})$ may be seen as being the sum of two components, E_u corresponding to 1s in $\mathbf{X}$ that are 0s in $\mathbf{S} \otimes \mathbf{F}$ (uncovered) and E_o corresponding to 0s in $\mathbf{X}$ that are 1s in $\mathbf{S} \otimes \mathbf{F}$ (overcovered):

$$E(\mathbf{X}, \mathbf{S} \otimes \mathbf{F}) = E_u(\mathbf{X}, \mathbf{S} \otimes \mathbf{F}) + E_o(\mathbf{X}, \mathbf{S} \otimes \mathbf{F})$$

with

$$E_u(\mathbf{X}, \mathbf{S} \otimes \mathbf{F}) = |\{\langle i, j\rangle | x_{ij} = 1, (\mathbf{S} \otimes \mathbf{F})_{ij} = 0\}|,$$

$$E_o(\mathbf{X}, \mathbf{S} \otimes \mathbf{F}) = |\{\langle i, j\rangle | x_{ij} = 0, (\mathbf{S} \otimes \mathbf{F})_{ij} = 1\}|.$$

Note that in the BMDP manual on the M8 procedure [1], E_u and E_o are called the positive and negative discrepancy, respectively. As a measure of coverage quality, we use

$$Q(\mathbf{X}, \mathbf{S} \otimes \mathbf{F}) = 1 - \frac{E(\mathbf{X}, \mathbf{S} \otimes \mathbf{F})}{||\mathbf{X}||},$$

which may be thought of as measuring coverage quality. Clearly, $Q(\mathbf{X}, \mathbf{S} \otimes \mathbf{F}) = 1$ if and only if $\mathbf{X} = \mathbf{S} \otimes \mathbf{F}$ (exact decomposition). Furthermore, $Q(\mathbf{X}, \mathbf{S} \otimes \mathbf{F})$ decreases with increasing error, i.e. with increasing $E(\mathbf{X}, \mathbf{S} \otimes \mathbf{F})$.

4. The Datasets Used

We compared the efficiency of the described methods using the artificial signals which are random mixtures of horizontal and vertical bars (the bars problem [6]) as well as the Mushroom dataset taken from the UCI Machine Learning Repository [2].

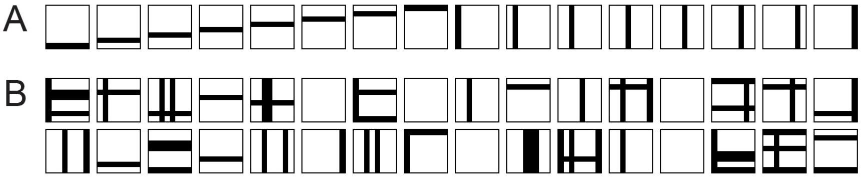

Fig. 1 A *Sixteen vertical and horizontal bars in 8-by-8 pixel images.* **B** *Examples of images in the standard bars problem. Each image contains two bars on average.*

4.1 The bars problem

The Bars Problem (BP) was introduced in [6] and has been considered in many papers in various modifications (see [15] for references) as a benchmark for learning of objects from complex patterns. In this problem, each pattern of the dataset is n-by-n binary pixel image containing several of $L = 2n$ possible (one-pixel wide) bars (Fig. 1). Pixels belonging and not belonging to the bar take values 1 and 0, respectively. For each image each bar could be chosen with the probability C/L, where C is the mean number of bars mixed in an image. In the point of intersection of vertical and horizontal bars, pixel takes the value 1. The Boolean summation of pixels belonging to different bars simulates the occlusion of objects. The task is to recognize all bars as individual objects on the basis of a dataset containing M images consisting of bar mixtures. In most papers where BP was used as a benchmark C was set to 2 and $n = 8$.

In terms of BFA, bars are factors. The factor loadings $f_{ij}, j = 1, \ldots, N$ take the value One for the pixels constituting the ith bar and the value Zero for the pixels not constituting it, $N = n^2$ is the total number of pixels in an image. Each image is Boolean superposition of factors. Factor scores take values One or Zero dependently on bar presence or absence in the image. Thus, the bars problem is a special case of BFA. We consider the case of homogeneously distributed image noise both in the form of common factor distortion and in the form of specific factors. Particularly, we put $p_{ij} = pf_{ij}$ and $q_j = q$. This means that the pixels constituting common factors (bars) can take Zero with equal probabilities p and the pixels can take One with equal probabilities q due to specific factors.

4.2 The Mushroom dataset

The Mushroom dataset consists of 8125 objects and 23 nominal attributes (for example, attribute “class” with values “edible” and “poisonous”, or attribute “cap-shape” taking values such as “bell”, “conical” or “convex”). We transformed this dataset to a Boolean matrix by nominal scaling, i.e. by replacing a nominal attribute y with k values $v_1, \ldots, v_k$ by k Boolean attributes $y_{v_1}, \ldots, y_{v_k}$ in such a way that at the ith row, the value of the column corresponding to y_{v_j} is 1 if and only if the value of the attribute y at the ith row in the original dataset is equal to v_j.

5. Experiments

In this section, we compare the efficiency of the three methods for Boolean factor analysis and Boolean matrix factorization. These methods are compared according to two criteria: information gain G and coverage quality Q. Initially, the methods are compared in solving BP. We used $n = 8$, $M = 800$, $C = 2$ and $L < 40$ then the execution time according to (2) and (7) amounted to less than a second for BVA2 and M8 and less than a minute for EMBFA.

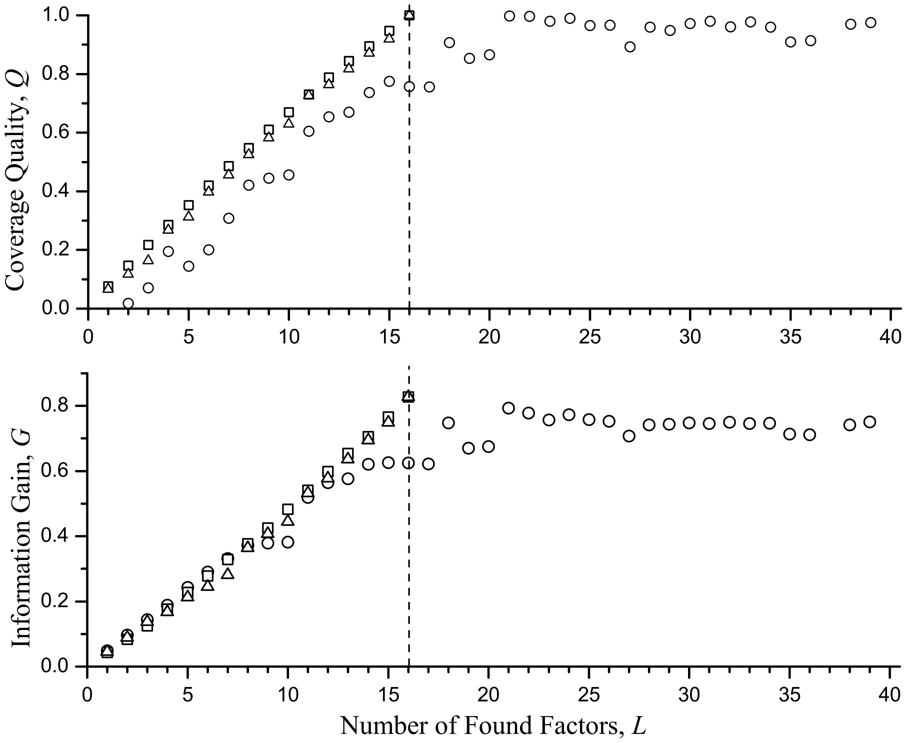

Fig. 2 *Dependencies of information gain G and coverage quality Q on the number of found factors for the case when noise is absent ($p = 1$ and $q = 0$). ○ – EMBFA, □ – BVA2, △ – M8.*

The dependency of G and Q on the number of found factors for the case when noise is absent ($p = 1$ and $q = 0$) is shown in Fig. 2. For all methods both criteria initially grow proportionally to the number of found factors. Since factors are assumed to be uniformly distributed in the dataset with equal probabilities $\pi_i = 1/8$, each new found factor provides a fixed increment to both indices G and Q. When all 16 bars are found, BVA2 and M8 stop because complete coverage is achieved. In this case, G and Q reach their maxima which correspond to the exact solution of BP. In EMBFA, as well as in M8, the total number of searched factors L must be assigned in advance as an input parameter. As Fig. 2 shows, EMBFA provides the maximal values of G and Q only at $L = 21$, that is when the assigned number of factors exceeds the actual number of factors (which is the number of bars). In this case, besides the bars some false factors (not bars) are found. As a result, EMBFA slightly loses to BVA2 and M8 in information gain. Another and more important cause of its loss is the omission of some factors in the observations of the dataset. Recall that this EMBFA implementation is supposing that not more than three factors are mixed in a pattern. However, in the generative model under consideration, the number of mixed factors K has the binomial distribution $B(K, C/L, L)$, where $C = 2$ and $L = 16$. According to this distribution, 13% of patterns containing more than three factors are generated. Since only three factors can be identified in these patterns, the remaining 9% of the onevalued scores are expected to be identified as zerovalued scores. The portion of onevalued scores missed by EMBFA amounts to 10%, which is close to the theoretical estimation obtained.

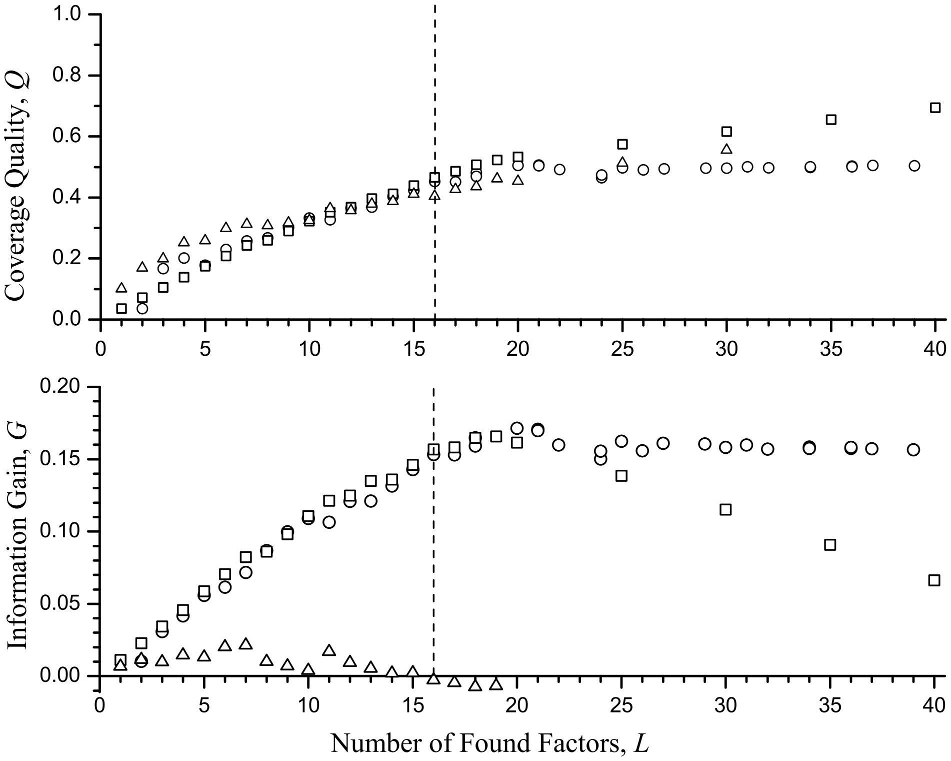

Fig. 3 *Dependencies of information gain G and coverage quality Q on the number of found factors for the case when common factors were not distorted ($p = 1$) and specific factors were present ($q = 0.3$). ○ – EMBFA, □ – BVA2, △ – M8.*

Fig. 3 illustrates the quality of the methods for the case when common factors were undistorted ($p = 1$) but specific factors were added ($q = 0.3$). When the number of the found factors L is less than the actual number of bars, for all methods coverage quality again increases almost proportionally to the number of found factors. For EMBFA and BVA2, this is accompanied by an increase of G but for M8 G remains to be close to zero. This is explained by the fact that EMBFA and BVA2 found mostly true bars but M8 found mostly false factors (not bars). For EMBFA gain reaches maximum at the point when the assigned number of searched factors L slightly exceeds the number of true factors. As mentioned above, this occurs because all true factors can be revealed by this method only when L exceeds the actual number of true factors. For BVA2 gain reaches maximum when the number of (correctly) found factors L equals to the number of bars. When L increases gain decreases due to the influence of found false factors. Recall that in BVA2 and M8 there is no notion of specific factors, that is why specific factors that appeared in observations at $q = 0.3$ were treated as common factors. For EMBFA, when the number of assigned factors L continues to increase, information gain does not decrease because specific factors are treated correctly.

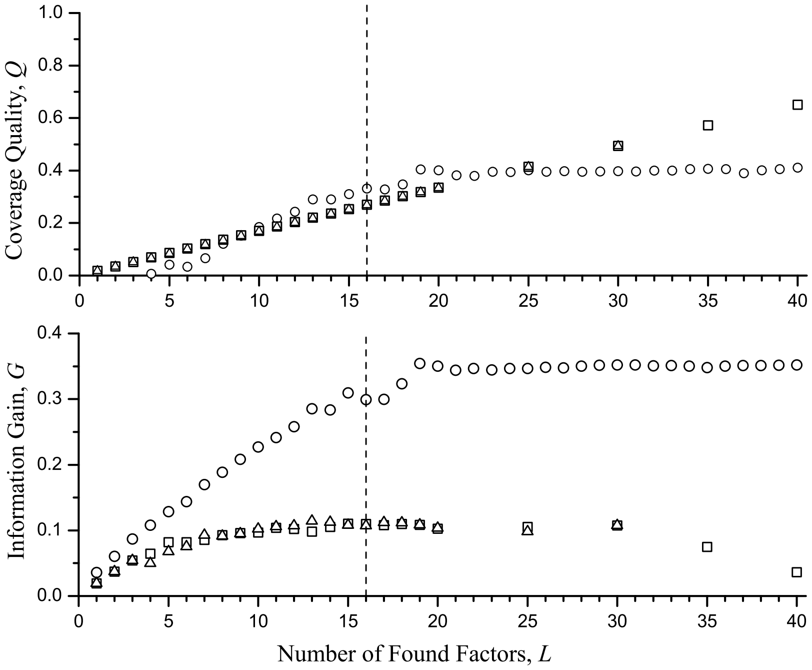

Fig. 4 *Dependency of gain G and coverage Q on the number of found factors for the case when common factors were distorted ($p = 0.6$) and specific factors were absent ($q = 0$). ○ – EMBFA, □ – BVA2, △ – M8.*

Fig. 4 illustrates the quality of the methods for the opposite case, that is when common factors were distorted ($p = 0.6$) but specific factors were absent ($q = 0$). In this case, both BVA2 and M8 perform much worse than EMBFA in information gain for all numbers of found factors. It means that most of the factors found by these methods are false. As for the previous cases, all true factors were found by EMBFA when the number of assigned factors L slightly exceeds their actual number. When the number of found factors continues to increase, information gain decreases for both BVA2 and M8 but not for EMBFA. In contrast, coverage quality continues to increase for BVA2 and M8 but not for EMBFA. However, coverage quality increases in this case only on account of false factors and, therefore, it does not provide useful information on the latent regular structure of the dataset. Thus, only performance of EMBFA happened to be insensitive to this kind of noise in the input data.

Note that for $L = 19$, when both indices Q and G for EMBFA reach maxima, coverage quality for EMBFA exceeds those for both BVA2 and M8 despite the claim of [4, 16] that those methods maximize Q for any given L. This gives a good example that greedy algorithms used by these methods provide not optimal solutions when greedy algorithms used by these methods actually provide reasonable but not optimal solutions. Note also that sometimes (in 5% of cases) EMBFA converges for a few steps to zero solution (as shown in Fig. 4, in our computation it happened in case of $L = 4$). To overcome this problem it is enough to restart it with another seed of initial values p_{ij}.

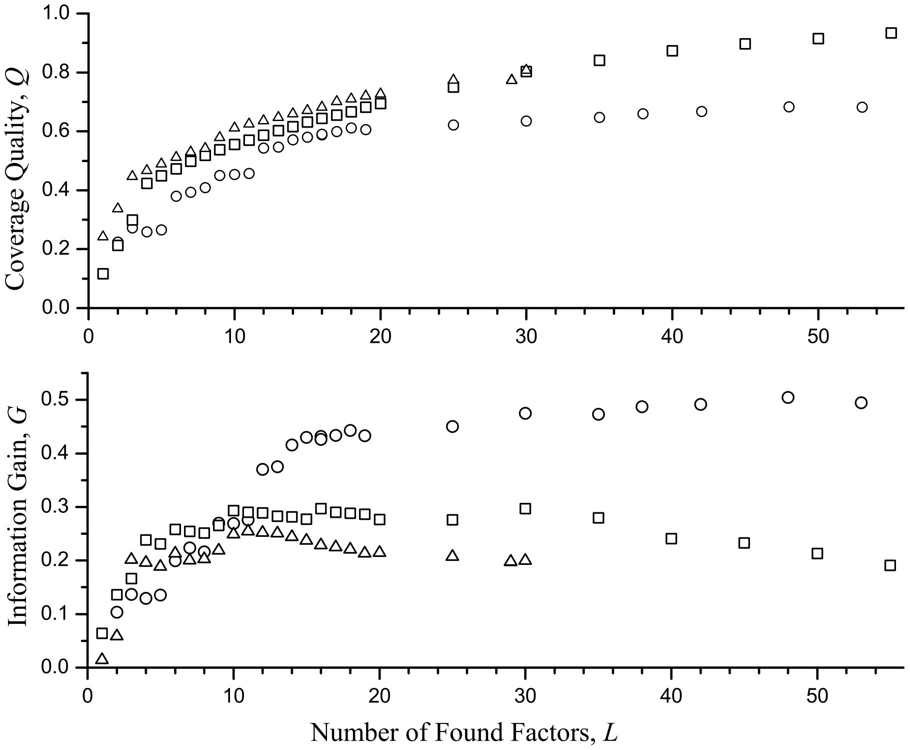

Fig. 5 *Dependency of gain G and coverage Q on the number of found factors is depicted for Mushroom dataset. ○ – EMBFA, □ – BVA2, △ – M8.*

The performance of the methods applied to the Mushroom dataset is shown in Fig. 5. All the methods demonstrate similar results until the number of found factors L is less than 10. If the methods attempt to assign more factors, the information gain obtained by BVA2 and M8 decreases, but for EMBFA it continues to increase monotonically. The drop of the increasing rate at $L = 20$ may be interpreted as the end of finding true factors. Following the results obtained for the bars problem one can speculate that the increase of coverage quality provided by BVA2 and M8 for $L > 10$ is explained only by the false factors.

6. Discussion

Performance of the BMF (BVA2 and M8) and BFA (EMBFA) methods was compared for the artificial dataset (the bars problem) and for the real datasets (the Mushroom dataset). Since for the artificial dataset its hidden factor structure is known in advance this information can be used for an exact evaluation of the method performance. For the bars problem we showed that both the BVA2 and M8 methods are perfect in solving BP when noise in the signals is absent. This is not surprising because in this case the search for factors is reduced to the dataset Boolean factorization, and both these methods were developed specially to perform this task. In contrast, EMBFA was developed for the Boolean factor analysis based on the given statistical generative model. As a result, EMBFA slightly loses to BVA2 and M8 in performance when noise-free signals are analyzed. On the contrary, both BVA2 and M8 lose to EMBFA when input signals are noisy. M8 is not able to reveal the hidden factor structure of the dataset for both kinds of noise, i.e. for common factor distortion and for appearance of specific factors, whereas BVA2 is unable to do this only for factor distortion. For appearance of specific factors it is able to find all true factors as EMBFA (3).

To estimate performance we used two indices: information gain G and coverage quality Q. The first index relates to BFA and the second one to BMF. As our experiments with the artificially generated data of the bars problem show, as the number of found factors L increases, sometimes these indices change identically, sometimes oppositely. Increasing of both Q and G corresponds to the case when a method finds true factors, whereas decreasing or constant G corresponds to the case when a method finds false factors. This gives a heuristic criterion for identification of true and false factors in real datasets. For BVA2 and M8 applied to the Mushroom dataset, both G and Q increase until L reaches 10. In this case, the factors found by all three methods are very similar. When $L > 10$, Q increases for all methods, but for EMBFA G increases and for BVA2 and M8 it decreases. This can be interpreted as finding true factors by EMBFA and false factors by BVA2 and M8. Note that methods of matrix factorization have no "false factor" notion and, therefore, have no grounded criteria for factor search termination. Contrary to this, in BFA such a criterion exists. It relates to the maximum of information gain: it is reasonable to continue the procedure of factors search until G increases and stop it when G reaches maximum or ceases to increase. For the Mushroom dataset G increases until $L = 20$. Thus, one can expect that EMBFA found 20 factors whereas BVA2 and M8 found only 10 factors suitable for further analysis by experts.

In spite of the fact that both BVA2 and M8 were developed to maximize coverage quality for every given number of factors during the run of algorithm, EMBFA is comparable with these methods even in this index. Thus, EMBFA could also be used for the Boolean matrix factorization. The disadvantage of EMBFA is that sometimes it converges to unreasonable zero solution. But this problem can be easily overcome by restart with another initial seed of parameters.

We have demonstrated the performance of the BMF and BFA approaches using only three methods as examples. However, we believe that the results obtained display the inherent differences of those two approaches. Although both approaches are aimed to reveal the hidden factor structure of a given dataset, BMF solves this problem directly by the best approximation of the dataset itself whereas BFA solves it statistically by searching the best correspondence between the dataset and some given probabilistic generative model. Thus, the advantage of the used BMF methods over the BFA method can be expected for noiseless data. Since the generative model underlying natural dataset and the noise level are usually unknown, it is most reasonable to apply both approaches to its analysis and to compare the results in terms of both information gain and coverage quality.

Acknowledgment

This research has been partly funded by project GACR P202/10/0262, by the IT4Innovations Centre of Excellence project, reg. no. CZ.1.05/1.1.00/02.0070 supported by Operational Programme 'Research and Development for Innovations' funded by the Structural Funds of the European Union and the state budget of the Czech Republic and by long-term strategic development financing of the Institute of Computer Science (RVO:67985807).

References

[1] Boolean Factor Analysis (8M) [online]: `http://www.statistical-solutions-software.com/bmdp-statistical-software/boolean-factor-analysis`

[2] Asuncion A., Newman D. J.: UCI Machine Learning Repository. University of California, Irvine, School of Information and Computer Sciences, 2007.

[3] Bartholomew D., Knott D. J.: Latent Variable Models and Factor Analysis. A Unified Approach. 3rd Edition. John Wiley & Sons, 2011.

[4] Belohlavek R., Vychodil V.: Discovery of optimal factors in binary data via a novel method of matrix decomposition. Journal of Computer and System Sciences, **76**, 1, 2010, pp. 3–20.

[5] Dempster A. P., Laird N. M., Rubin D. B.: Maximum likelihood from incomplete data via the EM algorithm. Journal of the Royal Statistical Society. Series B (Methodological), **39**, 1, 1977, pp. 1–38.

[6] Foldiak P.: Forming sparse representations by local anti-hebbian learning. Biological Cybernetics, **64**, 1990, pp. 165–170.

[7] Frank M., Streich A. P., Basin D., Bushmann. J. M.: Multi-assignment clustering for Boolean data. Journal of Machine Learning Research, **13**, 3, 2012, pp. 459–489.

[8] Frolov A., Husek D., Polyakov P.: Estimation of Boolean Factor Analysis Performance by Informational Gain. In: Snasel V., Szczepaniak P. S., Abraham A., Kacprzyk J., Eds., Advances in Intelligent Web Mastering-2, Springer 2010, pp. 83–94. Proceedings of the 6th Atlantic Web Intelligence Conference, Prague, 9. 9. 2009 – 11. 9. 2009.

[9] Frolov A., Husek D., Polyakov P.: New measure of Boolean factor analysis quality. In: Adaptive and Natural Computing Algorithms, A. Dobnikar, U. Lotric, and B. Ster, Eds., vol. **6593** of Lecture Notes in Computer Science. Springer, 2011, pp. 100–109.

[10] Frolov A. A., Husek D., Polyakov P. Y.: Boolean factor analysis by expectation-maximization method. In: Proceedings of the Third International Conference on Intelligent Human Computer Interaction (IHCI 2011), Prague, Czech Republic, August, 2011, M. Kudelka, J. Pokorny, V. Snasel, and A. Abraham, Eds., vol. 179 of Advances in Intelligent Systems and Computing. Springer, 2013, pp. 243–254.

[11] Ganter B., Wille R.: Formal concept analysis. Springer Berlin, 2013.

[12] Golub G., Kahan W.: Calculating the singular values and pseudo-inverse of a matrix. Journal of the Society for Industrial and Applied Mathematics: Series B, Numerical Analysis, **2**, 2, 1965, pp. 205–224.

[13] Lawly D. N., Maxwell A. E.: Factor Analysis as a Statistical Method. 2nd edition. Butterworth, London, 1971.

[14] Lee D., Seung H., et al.: Learning the parts of objects by non-negative matrix factorization. Nature, **401**, 6755, 1999, pp. 788–791.

[15] Lücke J., Sahani M.: Maximal causes for non-linear component extraction. Journal of Machine Learning Research, **9**, 2008, pp. 1227–1267.

[16] Mickey M. R., Mundle P., Engelman L.: Boolean factor analysis. In BMDP Statistical Software, W. Dixon, Ed. University of California Press, Berkeley, CA, 1983, pp. 538–545.

[17] Miettinen P., Mielikainen T., Gionis A., Das G., Mannila H.: The discrete basis problem. IEEE Transactions on Knowledge and Data Engineering, **20**, 10, 2008, 1348–1362.

[18] Paatero P., Tapper U.: Positive matrix factorization: A non-negative factor model with optimal utilization of error estimates of data values. Environmetrics, **5**, 2, 1994, pp. 111–126.

[19] Shingliar T., Hauskrecht M.: Noisy-OR component analysis and its application to link analysis. Journal of Machine Learning Research, **7**, 2006, pp. 2189–2213.