MEBF: a fast and efficient Boolean matrix factorization method

ABSTRACT

Boolean matrix factorization (BMF, or Boolean matrix decomposition) aims to find an approximation of a binary matrix as the Boolean product of two low rank sparse Boolean matrices, which could generate vast amount of information for the patterns of relationships between the features and samples. Inspired by binary matrix permutation theories and geometric segmentation, we developed a fast and efficient BMF approach called MEBF (Median Expansion for Boolean Factorization). Overall, MEBF adopted a heuristic approach to locate submatrices that are dense in 1s iteratively. At each iteration, MEBF permutates the rows and columns such that the permutated matrix is approximately upper triangular-like with closest simultaneous consecutive-ones property. A largest submatrix dense in 1 would be henceforth geometrically easy to reveal, which lies on the upper triangular area of the permutated matrix. MEBF demonstrated superior performances in higher coverage as well as more sparse basis than popular methods ASSO and PANDA+. Moreover, we demonstrate application of MEBF on a high dimensional single cell RNA sequencing reveals its further potential in dimension reduction and denoising on continuous matrix.

KEYWORDS

Boolean matrix factorization, Geometric segmentation, Binary matrix permutation

ACM Reference format:

. 2019. MEBF: a fast and efficient Boolean matrix factorization method. In *Proceedings of Beijing '19: ACM International Conference on Information and Knowledge Management , Beijing, China, November 03-07, 2019 (CIKM '19),* 4 pages.

DOI: 10.1145/1122445.1122456

1 INTRODUCTION AND BACKGROUND

Boolean matrix factorization (BMF) factorizes a binary matrix into approximately the product of two low rank binary matrices following Boolean algebra, and has shown its unique power in analyzing binary data [1, 2]. First discussed as a tiling problem in 1975 [3], BMF recently received wide attention after a series of work by Mittenin et al [1, 4]. Among these, the ASSO algorithm performs factorization by retrieving binary bases from row-wise correlation matrix in a heuristic manner [1]. Recently, an algorithm called nassua was developed by the same group[4]. Nassua optimizes the initialization of the matrix factorization by locating dense seeds embedded within the matrix, and with improved performance comparing to ASSO. However, optimal parameter selection remains a challenge for nassua. A second series of work was called PANDA developed by Claudio et al[5, 2]. PANDA aims to find the most significant patterns in the current binary matrix by discovering core patterns iteratively. After each iteration, PANDA only retains a residual matrix that removes all the non-zero values covered by identified patterns. PANDA+ was recently developed to reduce the noise level in core pattern detection and extension [2]. Aside from these two main lines, more algorithms and applications of BMF have been proposed in [6, 7]. In a most recent review [8], ASSO and PANDA+ are shown to achieve competitive performances, thus they act as benchmark methods in evaluating our methods.

CIKM '19, Beijing, China

978-1-4503-9999-9/18/06…$15.00

DOI: 10.1145/1122445.1122456

2 NOTATIONS AND PROBLEM STATEMENT

A matrix is denoted by a uppercase character with a superscript $n \times m$ indicating its dimension, such as $X^{n\times m}$, and with subscript $X_{i,:}, X_{:,j}, X_{ij}$ indicating ith row, jth column, or the $ijth$ element, respectively. A vector is denoted as a bold lowercase character with superscript representing its length, such as $\mathbf{a}^n$, and its subscript $\mathbf{a}_i$ indicates the ith element. A scalar value is represented by a lowercase character, such as a, and $[a]$ as its integer part. $|X|$ and $|\mathbf{x}|$ represent the L1 norm of a matrix and a vector. Under the Boolean algebra, the basic operations include $\wedge(AND, 1\wedge 1 = 1, , 1\wedge 0 = 0, 0\wedge 0 = 1)$, $\vee(OR, 1\vee 1 = 1, 0\vee 1 = 1, 0\vee 0 = 0)$, $\neg(NOT, \neg 1 = 0, \neg 0 = 1)$. Denote the Boolean element-wise operation sum, subtraction and product as $A \oplus B = A \vee B$, $A \ominus B = (A \wedge \neg B) \vee (\neg A \wedge B)$ and $A \circledast B = A \wedge B$, and the matrix product of two Boolean matrices as $X^{n\times m} = A^{n\times k} \otimes B^{k\times m}$, where $X_{ij} = \vee_{l=1}^{k} A_{li} \wedge B_{lj}$.

Given a binary matrix $X \in \{0,1\}^{n\times m}$ and a criteria parameter τ, the BMF problem is defined as identifying two binary matrix A^* and B^*, called pattern matrices, that minimize the cost function $\gamma(A, B; X)$ under criteria τ, i.e., $(A^*, B^*) = \text{argmin}_{A,B}(\gamma(A, B; X)|\tau)$. Here the parameter τ could vary with different problem assumptions. The criteria used in the study is to identify A^* and B^* with at most k patterns, i.e., $A \in 0, 1^{n\times k}$, $B \in 0, 1^{k\times m}$, and the cost function is $\gamma(A, B; X) = |X \ominus (A \otimes B)|$.

3 MEBF ALGORITHM FRAMEWORK

3.1 Motivation of MEBF

BMF is equivalent to decomposing the matrix into the sum of multiple rank 1 binary matrices, which is also referred as patterns in the BMF literature [5].

Lemma 1 (Submatrix detection). *Let A^*,B^* be the solution to* $\arg\min_{A^{n\times k}, B^{k\times m}} |X \ominus (A \otimes B)|$*, then the k patterns identified in A^*, B^* correspond to k submatrices in X that are dense in 1's. In other words, finding A^*, B^* is equivalent to identify submatrices X_{I_l,J_l}, $l = 1...k$, $I_l \subset (1,\ldots,n)$; $J_l \subset (1,\ldots,m)$ $s.t.|X_{I_l,J_l}| \geq t_0(I_l + J_l)$. Here $|I_l|$ is the cardinality of the index set I_l, t_0 is a positive number that controls the noise level of X_{I_l,J_l}.*

Proof. $\forall k$, it suffices to let I_l be the indices of A^*'s lth column, such that $A^*_{:,l} = 1$; and let J_l be the indices of B^*'s lth row such that $B^*_{l,:} = 1$. □

Motivated by Lemma 1, instead of looking for patterns directly, we turn to identify large submatrices in X that are enriched by 1, such that each set of row and column indices of the submatrices would indicate the locations of the non-zero values in each pattern.

Definition 1 (Direct consecutive-ones property, C1P). A binary matrix X has direct C1P if for each of its row vector, all 1s occur at consecutive indices.

Definition 2 (Simultaneous consecutive-ones property, SC1P). A binary matrix X has direct SC1P, if both X and X^T have direct C1P; and a binary matrix X has SC1P, if there exists a permutation of the rows and columns such that the permutated matrix has direct SC1P.

Definition 3 (Upper Triangular-Like matrix, UTL). An $m \times n$ binary matrix X is called an **U**pper **T**riangular-**L**ike (**UTL**) matrix, if 1) $\sum_{i=1}^{m} X_{i1} \leq \sum_{i=1}^{m} X_{i2} \leq \cdots \leq \sum_{i=1}^{m} X_{in}$; 2) $\sum_{j=1}^{m} X_{1j} \geq \sum_{j=1}^{m} X_{2j} \geq \cdots \geq \sum_{j=2}^{m} X_{nj}$ In other words, the matrix has non-increasing row sums, and non-decreasing column sums.

Lemma 2 (UTL matrix with direct SC1P). *If X is an UTL matrix and has direct SC1P, then an all'1 submatrix of the largest rectangular area in X has its row index of consecutive numbers $1, \cdots, i_0$, and column index of consecutive numbers $j_0, j_0 + 1, \cdots, n$. Specifically, i_0 and j_0 are to be found as illustrated in Figure 1.*

Definition 4 (Closest SC1P). Given a binary matrix X and a non-negative weight matrix W, a matrix $\hat{X}$ that has SC1P and minimizes the distance $d_W(X, \hat{X})$ is the closest SC1P matrix of X.

Based on Lemma 2, we could find all the submatrices in Lemma 1 by permutating rows and columns of matrix X to be an UTL matrix with closest direct SC1P, locating the its largest submatrix of all 1s;and reduce by repeating the process iteratively. However, making an UTL matrix may be easy, finding matrix of closest SC1P of matrix X is NP-hard.

Lemma 3 (Closest SC1P). *Given a binary matrix X and a non-negative weight matrix W , finding a matrix $\hat{X}$ that has SC1P and minimizes the distance $d_W(X, \hat{X})$ is an NP-hard problem.*

The NP-hardness of the closest SC1P problem has been shown in [9, 10]. Both exact and heuristic algorithms are known for the problem, and it has also been shown if the number of rows or columns is bounded, then solving closest SC1P requires only polynomial time [11]. In our MEBF algorithm, we attempt to address it by using heuristic methods and approximation algorithms.

3.2 Overview

In detail, for an Boolean matrix (Figure 2a), MEBF first rearranges the matrix to obtain an approximate UTL matrix with direct SC1P. This was achieved by reordering the rows so that the row norms are non-increasing, and reordering the columns so that the column norms are non-decreasing (Figure 2b). Then, MEBF takes either the column or row with medium number of 1s as one basis (Figure 2c). The pattern propagates to other columns or rows with

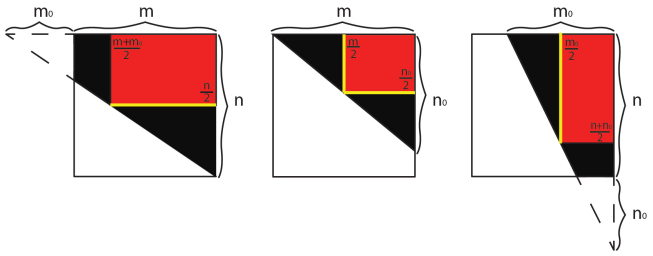

Figure 1. In an UTL matrix with direct SC1P, the all'1 submatrix with the largest rectangular area is the one defined by the rectangular on the upper triangular of the matrix, as shown in red. The width and height of the rectangulars are denoted by the two numbers inside the red colored ones. For ease of illustration, the matrix have no all-zero rows or columns. Back to Lemma 2, the i_0 corresponds to $\frac{n}{2}, \frac{n_0}{2}, \frac{n+n_0}{2}$; and j_0 corresponds to $\frac{m-m_0}{2}, m - m_0, m - m_0$, for the three simplified scenarios.

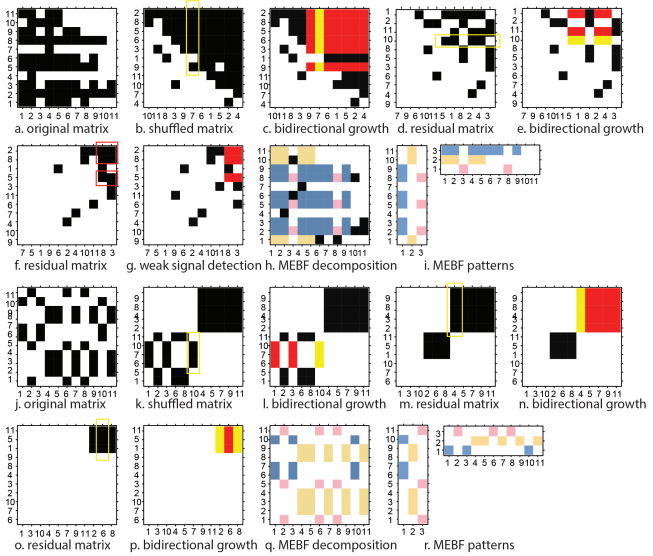

Figure 2: A schematic overview of the MEBF pipeline. (a-i) and (j-r) demonstrate two common scenarios where the matrix is roughly upper triangular and block diagonal.

a bidirectional growth algorithm until certain stopped criteria is met. Before the end of each iteration, MEBF computes a residual matrix by doing a Boolean subtraction of the rank-1 pattern matrix identified at this step (Figure 2d), and selects either the column or row pattern, which minimizes the cost function under the current residual matrix (Figure 2e). Repeat this procedure until the convergence criteria was hit. If the patterns identified by the bidirectional growth algorithm stopping deceasing the cost function before the convergence criteria was met, another step called weak signal detection was introduced to identify the best remaining pattern in the current residual matrix (Figure 2f,g). Figure 2j-2r illustrated a special case, where the original matrix is roughly block diagonal (Figure 2j). Through this process, it showed that MEBF may not be able to detect the largest pattern in the first attempt (Figure 2k,l), it still manages to identify the major patterns in relatively early

stages (Figure 2m,n) and achieves good factorization results (Figure 2o,p,q,r). The computational complexity of bidirectional growth and weak signal detection algorithms are both O(nm) and the complexity of each iteration of MEBF is O(nm). The main algorithm of MEBF is illustrated below:

Algorithm 1: MEBF

Inputs: $X \in \{0,1\}^{n\times m}$, $t \in (0,1)$,τ
Outputs: $A^* \in \{0,1\}^{n\times k}$, $B^* \in \{0,1\}^{k\times m}$
$MEBF(X,t,\tau)$:
$X_{residual} \leftarrow X$
while $!\tau$ **do**
  $(\mathbf{a},\mathbf{b}) \leftarrow bidirectional\_growth(X_{residual}, t)$
  if $\gamma(\boldsymbol{a},\boldsymbol{b};X_{residual}) < 0$ **then**
    $(\mathbf{a},\mathbf{b}) \leftarrow weak\_signal\_detection(X_{redidual}, t)$;
    if $\gamma(\boldsymbol{a},\boldsymbol{b};X_{residual}) < 0$ **then**
      break ;
  $A^* \leftarrow A.append(\mathbf{a})$
  $B^* \leftarrow B.append(\mathbf{b})$
  $X_{residual_{ij}} \leftarrow 0$ when $(\mathbf{a}\otimes\mathbf{b})_{ij} = 1$
end

3.3 Bidirectional Growth

For a binary matrix X, we first rearrange X as mentioned above. Denoted the rearranged X as X', the median column and median row of the rearranged matrix as $X'_{:,med}$ and $X'_{med,:}$ as shown in Figure 1. Denoted $X_{:,(med)}$ and $X_{(med),:}$as the column $X'_{:,med}$ and row $X'_{med,:}$ in the original matrix. The similarity between $X_{:,(med)}$ and columns of X can be computed as column wise correlation vector $\mathbf{m} \in (0,1)^m$, where $\mathbf{m}_i = \frac{<X_{:,i}, X_{:,(med)}>}{<X_{:,(med)}, X_{:,(med)}>}$. Similarly, the similarity between $X_{(med),:}$ and rows of X was similarly calculated and denoted as $\mathbf{n} \in (0,1)^n$. A pre-specified threshold $t \in (0,1)$ was further applied to filter out the rows and columns in rearranged X with strong similarity to the median column and row. In each iteration, we select the row or column pattern whichever achieving a smaller cost function.

Algorithm 2: Bidirectional Growth

Inputs: $X \in \{0,1\}^{n\times m}$, $t \in (0,1]$
Outputs: (**a**,**b**)
$bidirectional\_growth(X,t)$:
$X' \leftarrow X[order(rowSums), order(colSums)]$
$\mathbf{d} \leftarrow X_{:,(med)}, \mathbf{e} \leftarrow \{0,1\}^m$
$\mathbf{f} \leftarrow X_{(med),:}, \mathbf{g} \leftarrow \{0,1\}^n$
for j in $1...m$
  $\mathbf{m}_j = \frac{<X_{:,j},\mathbf{d}>}{<\mathbf{d},\mathbf{d}>}$, $\mathbf{e}_j = \mathbf{m}_j > t$
for i in $1...n$
  $\mathbf{n}_i = \frac{<X_{i,:},\mathbf{f}>}{<\mathbf{f},\mathbf{f}>}$, $\mathbf{g}_i = \mathbf{n}_i > t$
if $\gamma(\boldsymbol{e},\boldsymbol{d};X) > \gamma(\boldsymbol{f},\boldsymbol{g};X)$ **then**
  $\boldsymbol{a} \leftarrow \boldsymbol{d}$; $\boldsymbol{b} \leftarrow \boldsymbol{e}$;
else
  $\boldsymbol{a} \leftarrow \boldsymbol{f}$; $\boldsymbol{b} \leftarrow \boldsymbol{g}$;
end

3.4 Weak Signal Detection Algorithm

The patterns iteratively identified by the bidirectional growth algorithm cannot guarantee a constant decrease of the cost function, especially when the "large" patterns have been identified and the "small" patterns are easily confused with noise. To identify remaining weak patterns from a residual matrix, we set a week signal detection algorithm to locate the regions that may still have true patterns. Here, in our weak signal detection algorithm, we search the two columns with most 1s and the two rows with most 1s, and evaluate whether the cost function could be decreased or not by including a base generated from them.

Algorithm 3: Weak Signal Detection

Inputs: $X \in \{0,1\}^{n\times m}$, $t \in (0,1]$
Outputs: (**a**,**b**)
Weak_signal_detection(X,t)
$X' \leftarrow X[order(rowSums), order(colSums)]$
$\mathbf{d} \leftarrow X'_{1,:} \wedge X'_{2,:}$, $\mathbf{e} \leftarrow \{0,1\}^m$
$\mathbf{f} \leftarrow X'_{:,1} \wedge X'_{:,2}$, $\mathbf{g} \leftarrow \{0,1\}^n$
for j in $1...m$
  $\mathbf{m}_j = \frac{<X_{:,j},\mathbf{d}>}{<\mathbf{d},\mathbf{d}>}$, $\mathbf{e}_j = \mathbf{m}_j > t$
for i in $1...n$
  $\mathbf{n}_i = \frac{<X_{i,:},\mathbf{f}>}{<\mathbf{f},\mathbf{f}>}$, $\mathbf{g}_i = \mathbf{n}_i > t$
if $\gamma(\boldsymbol{e},\boldsymbol{d};X) > \gamma(\boldsymbol{f},\boldsymbol{g};X)$ **then**
  $\boldsymbol{a} \leftarrow \boldsymbol{d}$; $\boldsymbol{b} \leftarrow \boldsymbol{e}$;
else
  $\boldsymbol{a} \leftarrow \boldsymbol{f}$; $\boldsymbol{b} \leftarrow \boldsymbol{g}$;
end

4 EVALUATION ON SYNTHETIC DATA

We adopted a similar idea from a recent review to evaluate the performance of the algorithms [8]. We simulated binary matrix X by the Boolean product of pattern matrices U and V with different number of patterns, sparsity and noise levels. We evaluate how much the original noisy binary matrix and its true patterns can be recovered, with two measures, called coverage and sparsity:

$$Coverage := 1 - \frac{|(U\otimes V) \ominus (A^*\otimes B^*)|}{|U\otimes V|}$$

$$Sparsity := \frac{|A^{*n\times k}| + |B^{*k\times m}|}{(n+m)\times k}$$

Specifically, "coverage" measures how the two factorizing matrices could recover the true signals in the original matrix, and "sparsity" measures the level of parsimonies of the pattern matrices. A good factorization should have large coverage and small sparsity measures. To the best of our knowledge, the conditions to guarantee a unique solution of the BMF problem has not been theoretically derived, thus we do not directly compare the factorized and true pattern matrices directly, i.e., U vs A^*, and V vs B^*. Here A^* and B^* are the pattern matrices decomposed by the three different algorithms. We simulate binary matrices $X^{n\times m} = U^{n\times k} \otimes V^{k\times m} + E$, where each element of U and V follows an identical Bernoulli random variable. E_{ij} is also a Bernoulli random variable. The impact of the noise is through flipping the binary values of X_{ij} if $E_{ij} = 1$.

Let $n = m = 100, k = 5$, and the signal level and noise level parameters, p,p_0 each takes two different values, i.e. $p = 0.2, 0.4$;$p_0 = 0.01, 0.05$. We simulated such binary matrix for 10 times, each with four scenarios as shown in Figure 3. The performances of the three methods, MEBF, ASSO and PANDA+ are compared. It is noteworthy each algorithm requires one parameter as a standard input. For a comprehensive evaluation, we tested different parameter values, and to be fair, only the parameter with the best performance was used for the comparisonpicked. The convergence criteria were set as when (1) 10 patterns were identified; (2) if a newly identified pattern does not decrease the cost function.

As a result, each algorithms achieved consistent coverage measures of low variance under different parameter, suggesting robust performance of the methods. In all the four scenarios, MEBF demonstrated superior performances than the other two in terms of both coverage and sparsity, under different signal strengths and noise levels. As shown in Figure 3, when the number of patterns increases, MEBF (green) manages to achieve a higher coverage and overall sparsity scores compared with ASSO (orange) and PANDA+ (blue). Also, patterns identified in later stage by MEBF tend to be more and more sparse, indicating a more precise representation of such decomposition.

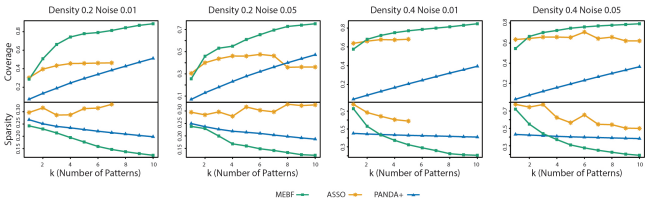

Figure 3. Performance evaluation on synthetic data

5 APPLICATION ON REAL DATA

Guided by Boolean operations, binary matrix factorization could be helpful in continuous matrix factorization, as it has been shown that the Boolean rank of a matrix could be much smaller than the regular matrix rank [12]. Recently, clustering of single cells using single-cell RNA sequencing data has gained extensive utilities in many fields, wherein the biggest challenge is that the high dimensional gene features, mostly noise features, makes the distance measure of single cells in clustering algorithm highly unreliable. We applied MEBF on a single cell RNA sequencing (scRNAseq) data [13], that measured more than 600 gene expression features for over 5,000 single cells, i.e., $X \in R^{600\times5,000}$. We first binarize original matrix $X \in R^{n\times m}$ to $X^* \in 0, 1^{n\times m}$, s.t. $X^*_{ij} = 1$ where $X_{ij} > 0$. Then, applying MEBF on X^* with parameters ($t = 0.6, k = 5$) outputs $A^{m\times k}, B^{k\times n}$. Let $X^{**} = A \otimes B$ and $X_{use} = X * X^{**}$. X_{use} represents a denoised version of X, by retaining only the entries in X pertinent to meaningful hidden structures extracted by MEBF. Compared with using X, clustering on X_{use} results in much tighter and well separated clusters, as visualized by t-SNE plots shown in Figure 4. t-SNE is an non-linear dimensional reduction approach for the visualization of high dimensional data[14]. It is worth noting that the heterogeneity among cell types with such a high dimensional feature space could be captured by matrices of Boolean rank equal to 5.

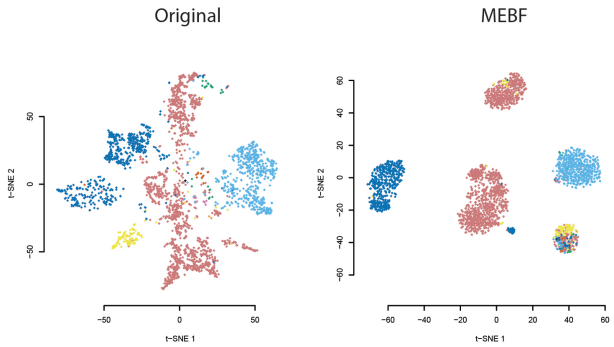

Figure 4. Visualization comparison of clustering on high dimensional data w/o and w MEBF de-noising

6 CONCLUSION

We developed a fast and efficient algorithm for binary matrix factorization, and adopted a heuristic approach to locate submatrices that are dense in 1s iteratively, where each such submatrix corresponds to one pattern in BMF. The submatrix identification was inspired by binary matrix permutation theory and geometric segmentation. Approximately, we permutate rows and columns of the input matrix so that the 1's could be "driven" to the upper triangular of the matrix as much as possible. Compared with two state of the art methods, ASSO and PANDA+, MEBF achieved better coverage and sparsity measures. Additionally, we used a single cell RNA-Seq expression data to demonstrate the potential of MEBF in denoising a continuous-valued matrix and greatly reduce its complexity.

REFERENCES

[1] Pauli Miettinen et al. "The discrete basis problem". In: *IEEE transactions on knowledge and data engineering* 20.10 (2008), pp. 1348–1362.

[2] Claudio Lucchese, Salvatore Orlando, and Raffaele Perego. "A Unifying Framework for Mining Approximate Top-k Binary Patterns". In: *IEEE Transactions on Knowledge and Data Engineering* 26.12 (2013), pp. 2900–2913.

[3] Larry J Stockmeyer. *The set basis problem is NP-complete.* IBM Thomas J. Watson Research Division, 1975.

[4] Sanjar Karaev, Pauli Miettinen, and Jilles Vreeken. "Getting to know the unknown unknowns: Destructive-noise resistant boolean matrix factorization". In: *Proceedings of the 2015 SIAM International Conference on Data Mining.* SIAM. 2015, pp. 325–333.

[5] Claudio Lucchese, Salvatore Orlando, and Raffaele Perego. "Mining top-k patterns from binary datasets in presence of noise". In: *Proceedings of the 2010 SIAM International Conference on Data Mining.* SIAM. 2010, pp. 165–176.

[6] Tammo Rukat et al. "Bayesian Boolean matrix factorisation". In: *Proceedings of the 34th International Conference on Machine Learning-Volume 70.* JMLR. org. 2017, pp. 2969–2978.

[7] Furkan Kocayusufoglu, Minh X Hoang, and Ambuj K Singh. "Summarizing Network Processes with Network-Constrained Boolean Matrix Factorization". In: *2018 IEEE International Conference on Data Mining (ICDM).* IEEE. 2018, pp. 237–246.

[8] Radim Belohlavek, Jan Outrata, and Martin Trnecka. "Toward quality assessment of Boolean matrix factorizations". In: *Information Sciences* 459 (2018), pp. 71–85.

[9] Esa Junttila et al. "Patterns in permuted binary matrices". In: (2011).

[10] Marcus Oswald and Gerhard Reinelt. "The simultaneous consecutive ones problem". In: *Theoretical Computer Science* 410.21-23 (2009), pp. 1986–1992.

[11] Marcus Oswald. "Weighted consecutive ones problems". PhD thesis. 2003.

[12] Art B Owen, Patrick O Perry, et al. "Bi-cross-validation of the SVD and the nonnegative matrix factorization". In: *The annals of applied statistics* 3.2 (2009), pp. 564–594.

[13] Sidharth V Puram et al. "Single-cell transcriptomic analysis of primary and metastatic tumor ecosystems in head and neck cancer". In: *Cell* 171.7 (2017), pp. 1611–1624.

[14] Laurens van der Maaten and Geoffrey Hinton. "Visualizing data using t-SNE". In: *Journal of machine learning research* 9.Nov (2008), pp. 2579–2605.