Binary Matrix Factorization with Applications

Zhongyuan Zhang
Academy of Math and Systems Science
Chinese Academy of Sciences
Beijing, 100080, P.R. China
zhzhyuan@amss.ac.cn

Tao Li
School of Computer Science
Florida International Univ.
Miami, FL 33199, USA
taoli@cs.fiu.edu

Chris Ding
Computer Science and Engineering Dept.
Univ. of Texas at Arlington
Arlington, TX 76019, USA
chqding@uta.edu

Xiangsun Zhang
Academy of Math and Systems Science
Chinese Academy of Sciences
Beijing, 100080, P.R. China
zxs@amt.ac.cn

Abstract

An interesting problem in Nonnegative Matrix Factorization (NMF) is to factorize the matrix X which is of some specific class, for example, binary matrix. In this paper, we extend the standard NMF to Binary Matrix Factorization (BMF for short): given a binary matrix X, we want to factorize X into two binary matrices W,H (thus conserving the most important integer property of the objective matrix X) satisfying $X \approx WH$. Two algorithms are studied and compared. These methods rely on a fundamental boundedness property of NMF which we propose and prove. This new property also provides a natural normalization scheme that eliminates the bias of factor matrices. Experiments on both synthetic and real world datasets are conducted to show the competency and effectiveness of BMF.

1. Introduction

1.1. Binary Data Clustering

Binary data have been occupying a special place in the domain of data analysis [15]. Typical applications for binary data include market basket data, document-term data, Web click-stream data (users vs websites), DNA microarray expression profiles, or protein-protein complex interaction network. For instance, in market basket data, each data transaction can be represented as a binary vector where each element indicates whether or not any of the corresponding item/product was purchased. In document data, each document can be represented as a binary vector where each element indicates whether a given word/term was present or not.

A distinctive characteristic of the binary data is that the features (attributes) they include have the same nature as the data they intend to account for: both are binary. This characteristic implies the symmetric association relations between the data and features. In addition, binary data are usually high dimensional. Most clustering algorithms do not work efficiently in high dimensional spaces due to the *curse of dimensionality*. Many feature selection techniques have been applied to reduce the dimensionality. However, the correlations among the dimensions are often specific to data locality; in other words, some data points are correlated with a given set of features and others are correlated with respect to different features [1]. For example, a subset of users are responsible for a subset of websites; a subset of genes are expressed under a subset of conditions. Thus for binary data clustering, one has to combine feature selection and clustering together.

1.2. Binary Matrix Factorization (BMF)

Binary data matrix is a special case of non-negative matrix. The nonnegative matrix factorization (NMF) has been shown recently to be useful for clustering [16]. NMF can be traced back to 1970s (Notes from G. Golub) and is studied extensively by Paatero [22]. The work of Lee and Seung [13, 14] brought much attention to NMF in machine learning and data mining fields. A recent theoretical analysis [4] shows the equivalence of NMF and spectral clustering and

K-means clustering. Various extensions and variations of NMF have been proposed recently [5, 12, 18, 24, 27].

Despite significant research progress in NMF, few attempts have been made to extend the standard NMF methods to binary matrices. As we discussed in Section 1.1, binary data has many distinctive characteristics and we believe these new aspects of binary data deserve theories and algorithms of their own. An attempt of binary matrix factorization (e.g., block diagonal clustering) was proposed in [15] where the input binary data matrix X is decomposed into a binary matrix W and a non-negative matrix H. In block diagonal clustering, the binary matrix W explicitly designates the cluster memberships for data points and the matrix H indicates the feature representations of each cluster. In this paper, we study the problem of decomposing the input binary data matrix X into two binary matrices W and H. The binary matrices W and H preserve the most important integer property of the input matrix X and they also explicitly designates the cluster memberships for data points and features. We call the decomposition as Binary Matrix Factorization, denoted by BMF.

1.3. Content of the Paper

Both the theoretical and practical aspects of BMF are studied in the paper. This rest of paper is organized as follows: Section 2 introduces the notations and gives an overview on matrix factorization methods related to BMF; Section 3 proves the boundedness property of nonnegative matrix factorizations and discusses the theoretical foundations for BMF; Section 4 presents penalty function and thresholding algorithms for BMF; Section 5 illustrates BMF using numerical examples; Section 6 shows experimental results on both synthetic and real world datasets; and finally Section 7 concludes.

2. Related Matrix Factorization Algorithms

Let the input data matrix $X = (\mathbf{x}_1, \cdots, \mathbf{x}_n)$ contain the collection of n data column vectors. Generally, we factorize X into two matrices,

$$X \approx WH, \tag{1}$$

where $X \in \mathbb{R}^{p\times n}$, $W \in \mathbb{R}^{p\times r}$ and $H \in \mathbb{R}^{r\times n}$. Generally, the rank of matrices W,H is much lower than the rank of X (i.e., $r \ll \min(p,n)$). Here we provide an overview on the matrix factorization methods related to BMF:

1. **SVD:** The classic matrix factorization is Principal Component Analysis (PCA) which uses the Singular Value Decomposition (SVD) [6, 8], $X \approx U\Sigma V^T$, where we allow U,V to have mixed-signs; the input data could have mixed-signs. absorbing Σ into U, we can write
$$\text{SVD:} \quad X_\pm \approx U_\pm V_\pm. \tag{2}$$
2. **NMF:** When the input data is nonnegative, and we restrict F and G to be nonnegative. The standard NMF can be written as
$$\text{NMF:} \quad X_+ \approx W_+ H_+ \tag{3}$$
using an intuitive notation for $X,W,H \geq 0$.
3. **GCM (General Clustering Model) for binary data:** When the input data X is binary, a general matrix factorization model is $X_{0-1} \approx W_{0-1}G_+H_+$ where W is binary and G specify the cluster associations between data points and features [15]. If we absorb G_+ into H_+, the model can be represented as
$$\text{GCM:} \quad X_{0-1} \approx W_{0-1}H_+. \tag{4}$$
Note that if G is an identity matrix, then after appropriate permutation of the rows and columns, the approximation data take the form of a block diagonal matrix.
4. **BMF:** The BMF can be written as
$$\text{BMF:} \quad X_{0-1} \approx W_{0-1}H_{0-1}. \tag{5}$$
We note that boolean factorization [21]
$$X_{0-1} \approx W_{0-1} \oplus H_{0-1}$$
has also been considered.

3. Boundedness Property of Nonnegative Matrix Factorization

3.1. Two Problems

BMF can be derived based on variants of standard NMF algorithms. However, two problems need to be resolved first.

- **Uniqueness**. For any solution W, H, we can always find a diagonal matrix $D \geq 0$ such that $(WD)(D^{-1}H) = (WH)$, i.e., the solution for W,H is not unique.
- **Scale**. Another related problem is the scale problem when discretizing W, H into binary matrices. A standard discretization method sets $W_{ij} = 0$ or 1 based on a threshold W_0. We write as $W_{ij} \leftarrow \theta(W_{ij} - W_0)$, where the Heaviside step function is defined as
$$\theta(x) = \begin{cases} 1 & x \geqslant 0 \\ 0 & x < 0. \end{cases}$$
Clearly, starting with W in different scales, the final discretization results would be different.

The boundedness property of NMF discussed below is motivated by the above two problems.

3.2. Boundedness Property

In this section, we propose and prove a new fundamental property of NMF. A standard decomposition of matrix is SVD: $X = U\Sigma V^T = U'V'$ where $U' = U\Sigma^{1/2}$ and $V' = V\Sigma^{1/2}$ typically contain mixed sign elements. NMF differs from SVD due to the absence of cancellation of plus and minus signs. But what is the fundamental signature of this absence of cancellation?

Let $X = WH$, where $X \in \mathbb{R}^{p\times n}$, $W \in \mathbb{R}^{p\times r}$ and $H \in \mathbb{R}^{r\times n}$. We say the input matrix X is bounded, if $0 \le X_{ij} \le 1$. Note that binary input matrix is bounded and we can rescale a nonnegative input matrix into the bounded form. The boundedness property of NMF states: *if X is bounded, then the factor matrices W, H must also be bounded, i.e.*, $0 \le W_{ik} \le 1$, $0 \le H_{kj} \le 1$.

We note that SVD decomposition does not has the boundedness property. In this case, even if the input data are in the range of $0 \le X_{ij} \le 1$, we can find some elements of U' and V' such that $U' > 1$ and $V' > 1$. For NMF, we motivate the property as following. We can write $X_{ij} = \sum_{k=1}^{r} W_{ik}H_{kj}$. Due to non-negativity, no cancellation can occur. Since $X_{ij} \le 1$, we must have $W_{ik}H_{kj} \le 1$. A sufficient condition for this to hold is the bounded property of elements of W and H, i.e., $W_{ik} \le 1, H_{kj} \le 1$. We prove in Theorem 1 that this is indeed the case.

Theorem 1 (Boundedness Theorem)
Let $0 \le X \le 1$ be the input data matrix. W, H are the nonnegative matrices satisfying

$$X = WH \tag{6}$$

There exists a diagonal matrix $D \ge 0$ such that

$$X = WH = (WD)(D^{-1}H) = W^*H^* \tag{7}$$

with

$$0 \le W^*_{ij} \le 1, 0 \le H^*_{ij} \le 1 \tag{8}$$

If X is symmetric and $W = H^T$, then $H^* = H$.

Proof: First of all, rewrite $W = (w_1, w_2, \cdots, w_r), H = (h_1, h_2, \cdots, h_r)^T$. Let

$$\begin{aligned} D_W &= diag(\max(w_1), \max(w_2), \cdots, \max(w_r)) \\ D_H &= diag(\max(h_1), \max(h_2), \cdots, \max(h_r)) \end{aligned}$$

where $\max(w_i), 1 \le i \le r$ is the largest element of the i-th column of W and $\max(h_j), 1 \le j \le r$ is the largest element of the j-th row of H.

Note

$$D_W = D_W^{1/2}D_W^{1/2}, \qquad D_H = D_H^{1/2}D_H^{1/2}.$$

$$D_W^{-1} = D_W^{-1/2}D_W^{-1/2}, \qquad D_H^{-1} = D_H^{-1/2}D_H^{-1/2}.$$

We obtain

$$\begin{aligned} X &= WH = (WD_W^{-1})(D_WD_H)(D_H^{-1}H) \\ &= (WD_W^{-1/2}D_H^{1/2})(D_H^{-1/2}D_W^{1/2}H). \end{aligned}$$

Construct D as $D = D_H^{-1/2}D_W^{1/2}$, then

$$W^* = WD_W^{-1/2}D_H^{1/2}, \qquad H^* = D_H^{-1/2}D_W^{1/2}H.$$

Thus Equation (7) is proved.

Furthermore,

$$\begin{aligned} (WD_W^{-1/2}D_H^{1/2})_{ij} &= W_{ij} \cdot \sqrt{\frac{\max(H_j)}{\max(W_j)}} \\ &= \frac{W_{ij}}{\max(W_j)} \cdot \sqrt{\max(W_j)\max(H_j)}. \end{aligned}$$

Without loss of generality, assuming that

$$\max(W_j) = W_{tj}, \qquad \max(H_j) = H_{jl},$$

then we have

$$\begin{aligned} \max(W_j) \cdot \max(H_j) &\le W_{t1}H_{1l} + \cdots W_{tj}H_{jl} + \cdots + W_{tr}H_{rl} \\ &= \sum_k W_{tk}H_{kl} = X_{tl} \le 1, \end{aligned}$$

So $0 \le W^*_{ij} \le 1$ and $0 \le H^*_{ij} \le 1$.
If X is symmetric and $W = H^T$,

$$H^*_{ij} = H_{ij} \cdot \sqrt{\frac{\max(H_i)}{\max(H_i)}} = H_{ij}.$$

which implies $H^* = H$.
□

Later we will show that we resolve the scale problem by applying the proposed rescaling in the theorem to the binary data as a special case of bounded matrices. In this way, discretization can work properly because W, H are in the same scales. This is crucial for our proposed penalty function algorithm and thresholding algorithm, making them more robust. The theorem can also be regarded as the normalization of W and H. Its function can be seen more clearly from Table 2.

4. Binary Matrix Factorization Algorithms

Given a matrix X composed of non-negative elements, the task of NMF is to factorize X into a non-negative matrix W and another non-negative matrix H such that $X \approx WH$. In general, the derived algorithm of NMF is as follows:

- Randomize W and H with positive number in [0, 1]. Select the cost function to be minimized.

- For W, update H, then update W for the updated H and so on until the process converges.

In this section, we extend the standard NMF to BMF: given a binary matrix X, we want to factorize X into two binary matrices W,H (thus conserving the most important integer property of the objective matrix X) satisfying $X \approx WH$. This is not straightforward and two parallel methodologies (e.g., penalty function algorithm and thresholding algorithm) have been studied and compared. we show that in this paper each of these two methods has its own advantages and disadvantages.

4.1. Penalty Function Algorithm

In terms of nonlinear programming, the problem of BMF can be represented as:

$$\begin{array}{ll} \min & J(W,H) = \sum_{i,j}(X_{ij} - (WH)_{ij})^2 \\ s.t. & H_{ij}^2 - H_{ij} = 0 \\ & W_{ij}^2 - W_{ij} = 0 \end{array}$$

which can be solved by a penalty function algorithm. The algorithm is described as follows:

Step 1: Initialize λ, W, H and ε.

Step 2: Normalize W, H using Theorem 1.

Step 3: For W and H, alternately solve:

$$\begin{aligned} \min J(W,H) &= \sum_{i,j}(X_{ij} - (WH)_{ij})^2 + \frac{1}{2}\lambda(H_{ij}^2 - H_{ij})^2 \\ &\quad + \frac{1}{2}\lambda(W_{ij}^2 - W_{ij})^2 \end{aligned}$$

Step 4: if $(H_{ij}^2 - H_{ij})^2 + (W_{ij}^2 - W_{ij})^2 < \varepsilon$
break
else
$\lambda = 10\lambda$, return to Step 3.
end

In step 1, W,H are initialized with the result of the original NMF algorithm [13, 14] applied to X. In step 3, the update rule is derived as follows : First, the derivative of the cost function $J(W,H)$ with respect to H is:

$$\frac{\partial}{\partial H_{au}} J(W,H) = -\sum_i (X_{iu} - (WH)_{iu}) W_{ia} + \lambda((2H_{au} - 1)(H_{au}^2 - H_{au})).$$

Let the step size

$$\alpha_{au} = H_{au} / (W^T(WH))_{au} + 2\lambda H_{au}^3 + \lambda H_{au}),$$

which is the longest stepsize that can maintain the non-negative property of H during the iterative process, then

$$\begin{aligned} H_{au} &= H_{au} - \alpha_{au} \frac{\partial}{\partial H_{au}} J(W,H) \\ &= H_{au}\left(\frac{(W^T X)_{au} + 3\lambda H_{au}^2}{(W^T WH)_{au} + 2\lambda H_{au}^3 + \lambda H_{au}}\right). \end{aligned}$$

By reversing the roles of the W and H, one can easily obtain the update rule of W. Similarly the update formula can be obtained when X is symmetric. The convergence of the algorithm is guaranteed as long as the minima of step 3 can be achieved.

4.2. Thresholding Algorithm

The second method is thresholding, in other words, finding the best thresholds w,h for W and H respectively so that the minima of the following problem can be achieved:

$$\min \quad F(w,h) = \frac{1}{2}\sum_{i,j}(X_{ij} - (\theta(W-w)\,\theta(H-h))_{ij})^2$$

where the Heaviside step function is defined as

$$\theta(x) = \begin{cases} 1 & x > 0 \\ 0 & x \le 0, \end{cases}$$

and $\theta(W)$ is element-wise operation: $\theta(W)$ is a matrix whose (i,j)-th element is $[\theta(W)]_{ij} = \theta(W_{ij})$. Initial values of W,H are given via the original NMF algorithm [13, 14].

As we can see, $\theta(x)$ is non-smooth, so the problem is a non-smooth optimization problem. There are two implementations to conquer this difficulty.

- Discretization Method: We discretize the domain $\{(w,h) : 0 \le w \le max(W), 0 \le h \le max(H)\}$ and try on every grid point to search for optimal thresholds (w^*, h^*).
- Gradient Descent Method: We approximate the Heaviside function by the function

$$\theta(x) \approx \phi(x) = \frac{1}{1 + e^{-\lambda x}}, \quad \lambda \textit{ is a large constant}.$$

In this paper, we use gradient descent method for optimization as the first implementation is too time-consuming. The gradient descent thresholding method can be described as follows:

Step 1: Initialize w_0, h_0, $k = 0$.

Step 2: Normalize W and H using Theorem 1.

Step 3: Compute gradient direction g_k of $F(w,h)$.

Step 4: Select the step size α_k.

Step 5: $w_{k+1} = w_{k+1} - \alpha_k g_k, h_{k+1} = h_k - \alpha_k g_k$.
if the stop criterion is satisfied,
$W = \theta(W - w_{k+1}); H = \theta(H - h_{k+1})$;*break*
else
k=k+1, return to step 3
end.

In step 3, the gradient direction g_k is:

$$\begin{aligned} g_k(1) &= \partial F(w,h)/\partial w \\ &= \partial \sum_{a,b} F(w,h)/\partial W^*_{ab} \cdot \partial W^*_{ab}/\partial w \\ &= \sum_{a,b}((XH^{*T})_{ab} - (W^*H^*H^{*T})_{ab}) \cdot \frac{e^{-\lambda(W^(?*)_{ab}-w)} \cdot \lambda}{(1+e^{-\lambda(W^*_{ab}-w)})^2}, \end{aligned}$$

$$\begin{aligned} g_k(2) &= \partial F(w,h)/\partial h \\ &= \partial \sum_{a,b} F(w,h)/\partial H^*_{ab} \cdot \partial H^*_{ab}/\partial h \\ &= \sum_{a,b}((W^{*T}X)_{ab} - (W^{*T}W^*H^*)_{ab}) \cdot \frac{e^{-\lambda(H^*_{ab}-h)} \cdot \lambda}{(1+e^{-\lambda(H^*_{ab}-h)})^2}. \end{aligned}$$

where $W^* = \phi(W - w_k), H^* = \phi(H - h_k)$.
In step 4, α_k can be selected by minimizing $F(w_k - \alpha_k g_k(1), h_k - \alpha_k g_k(2))$, but this is time-consuming. In practical, Wolfe line search method can be applied which requires α_k satisfying:

$$F(w_{k+1}, h_{k+1}) - F(w_k, h_k) \leq \delta\alpha_k g_k^T d_k,$$

$$g_{k+1}^T d_k \geq \sigma g_k^T d_k,$$

where $d_k = -g_k$ and δ, σ are constants, $0 < \delta < \sigma < 1$.

It can be proved that the step size α_k is well-defined in this way, that is, α_k exists as long as $g_k^T d_k < 0$. As a well studied method, gradient decent, the convergence is guaranteed.

5. Illustrative Examples

In this section, we use examples to: i) illustrate the detailed factorization results of BMF; ii) demonstrate the effects of normalization; iii) investigate the performance of the two BMF algorithms on the input matrices with different conditions such as sparsity parameters and ranks.

5.1. Two Simple Examples

First, we show the detailed results of BMF on two examples where the size of X is 5×8 and $r = 3$. Note that the two input matrices have different sparsity. In the following illustration, *Derivative* refers to the gradient decent thresholding method, *penalty* refers to the penalty function method, $Max = max_{ij}(|V - W \cdot H|_{ij})$, and $J(W,H)$ is the same as that in section 4.

Example 1: $\mathbf{X} = \begin{pmatrix} 1 & 0 & 1 & 0 & 1 & 0 & 1 & 1 \\ 1 & 1 & 0 & 1 & 1 & 1 & 0 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 0 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 0 & 1 & 1 & 1 & 0 & 1 \end{pmatrix}$.

Derivative:

$$W = \begin{pmatrix} 0 & 1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 1 \\ 1 & 0 & 0 \end{pmatrix}; \qquad \begin{aligned} Max &= 1 \\ J &= 1 \end{aligned}$$

$$H = \begin{pmatrix} 1 & 1 & 0 & 1 & 1 & 1 & 0 & 1 \\ 1 & 0 & 1 & 0 & 1 & 0 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \end{pmatrix};$$

Penalty:

$$W = \begin{pmatrix} 0 & 1 & 0 \\ 1 & 0 & 1 \\ 0 & 1 & 1 \\ 0 & 1 & 1 \\ 1 & 0 & 0 \end{pmatrix}; \qquad \begin{aligned} Max &= 1 \\ J &= 1 \end{aligned}$$

$$H = \begin{pmatrix} 1 & 1 & 0 & 1 & 1 & 1 & 0 & 1 \\ 1 & 0 & 1 & 0 & 1 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 & 0 & 1 & 0 & 0 \end{pmatrix};$$

Example 2: $\mathbf{X} = \begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 & 0 & 1 & 1 & 0 \\ 0 & 1 & 1 & 0 & 0 & 1 & 1 & 1 \\ 1 & 0 & 1 & 1 & 0 & 1 & 1 & 1 \\ 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 \end{pmatrix}$.

Derivative:

$$W = \begin{pmatrix} 0 & 0 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{pmatrix}; \qquad \begin{aligned} Max &= 1 \\ J &= 5 \end{aligned}$$

$$H = \begin{pmatrix} 1 & 0 & 0 & 1 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 & 1 & 1 & 1 \end{pmatrix};$$

Penalty:

$$W = \begin{pmatrix} 0 & 0 & 1 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 1 \\ 0 & 1 & 0 \end{pmatrix}; \qquad \begin{aligned} Max &= 1 \\ J &= 6 \end{aligned}$$

$$H = \begin{pmatrix} 0 & 0 & 0 & 1 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 & 1 & 1 \end{pmatrix}.$$

If we regard each column of X as a data point, then we can find that H matrices in both *Derivative* and *Penalty* preserve many important data characteristics of X (e.g., integer property, pair-wise distance).

5.2. Numerical Simulations

In this section, we perform a set of numerical simulations to examine the performance of the two BMF algorithms on the input matrices with different conditions and to demonstrate the effects of normalization. The input matrix X is generated as follows:

- **Step 1**. Randomize X with positive number in [0, 1].
- **Step 2**. for the element $X(i,j) > p$, $X(i,j) = 1$, otherwise $X(i,j) = 0$, where p is a pre-assigned parameter that controls the sparsity of X.

Table 1 shows the numerical results where the size of the input binary matrix X is 200×400. In Table 1, the density parameter P is selected from $\{0.2, 0.5, 0.8\}$. Note that NMF is a restricted form of matrix factorization. To evaluate the performance of NMF, we compare it with SVD using $||X - X^*||^2$ as the evaluation function where $X^* = W_{m,r} H_{r,n}$ via NMF and $X^* = \sigma_1 u_1 v_1^T + \sigma_2 u_2 v_2^T + \cdots + \sigma_r u_r v_r^T$ via SVD. *NMF* refers to the standard NMF algorithm, *BMF-penalty* refers to the penalty function method, and *BMF-threshold* refers to the gradient decent thresholding method. *Diff-W* and *Diff-H* show the significant difference between the results of penalty method and thresholding method which indicates that the two methods are equivalently important and can not be replaced by each other. From Table 1, we observe that when the input matrix X is dense (i.e., P is small), the penalty function algorithm works better than the thresholding algorithm and the thresholding algorithm is better when the input matrix X is sparse.

One useful consequence of Theorem 1 is the normalization of W, H, which eliminates the bias between W and H. This is especially true when the matrix X is sparse. Table 2 demonstrates the effect of normalization. P, again, refers to the sparsity parameter selected from $\{0.2, 0.5, 0.8\}$. The values in the bracket is the percentage of non-zero elements in the non-normalized case, and the values outside the bracket is the percentage in the normalized case. One can observe, from Table 2, that the normalization process has effectively eliminated the bias between W and H and made the results more robust. Without normalization, the resulting matrix H is often very sparse (sometimes it even becomes zero matrix) while W is very dense. As a result, much information that should be given via H is lost and this can not be compensated by the resulting dense matrix W.

	Sparsity(P)	W(%)	H(%)
r=3	0.2	33.3(33.3)	71.8(64.6)
	0.5	46(77.8)	29.2(9.8)
	0.8	6.7(67)	11.3(0)
r=5	0.2	40(78.8)	46.8(20)
	0.5	11.6(58.2)	37.4(0)
	0.8	27.1(46.1)	10(0)
r=10	0.2	29(54.9)	29.4(0.1)
	0.5	10(45.9)	17.2(0.03)
	0.8	17.8(36.5)	7.2(0)
r=20	0.2	13.1(45.4)	22.3(0)
	0.5	7.5(35.9)	12.7(0)
	0.8	5.2(26.5)	10.6(0)

Table 2. Comparison of the normalized case and non-normalized case (in parenthesis). Shown are percentage of nonzero elements.

6. Experimental Results

Three sets of experiments are conducted to evaluate the performance of BMF. First, Synthetic datasets is used to evaluate the effectiveness of BMF. A second set of experiments is performed on gene expression datasets to identify the bicluster structures. Finally BMF is applied on document datasets for document clustering.

6.1. Synthetic Datasets

6.1.1 Data Generation

We use the method described in [23] to generate synthetic datasets. Four datasets are generated with different bicluster structures as shown in Figure 1. The main advantage of using synthetic datasets is that the detailed bicluster structures are known and hence we can evaluate the performance of our BMF methods with different factors such as noise level and overlap degree systematically. In order to perform systematic evaluation with a large number of experiments, the

	P	SVD	NMF	BMF-penalty	BMF-threshold	Diff-W	Diff-H
r=3	0.2	1.2361	1.2361	1.6039	3.2744	0.04	0.0519
	0.5	1.9281	1.9287	3.7254	3.6519	0.0218	0.0186
	0.8	1.2391	1.2406	1.6299	1.6054	0.0033	0.0088
r=5	0.2	1.1970	1.1973	1.5923	4.1197	0.0426	0.0321
	0.5	1.8752	1.8784	3.8725	3.6280	0.0133	0.0147
	0.8	1.1889	1.1939	1.5770	1.5690	0.0249	0.0187
r=10	0.2	1.1208	1.1262	1.6025	4.3570	0.0520	0.0704
	0.5	1.7523	1.7736	3.9580	3.5345	0.0362	0.0275
	0.8	1.1203	1.1449	1.5950	1.5850	0.0333	0.0293
r=20	0.2	0.9803	1.0082	1.6099	4.4127	0.0706	0.1258
	0.5	1.5281	1.6086	3.4371	3.3909	0.0142	0.0273
	0.8	0.9799	1.0550	1.5567	1.5420	0.0169	0.0313

Table 1. Errors (in unit of 10^4) for various factorizations. Diff-W = root-mean-square difference between the BMF-penalty solution and BMF-threshold solution on W.

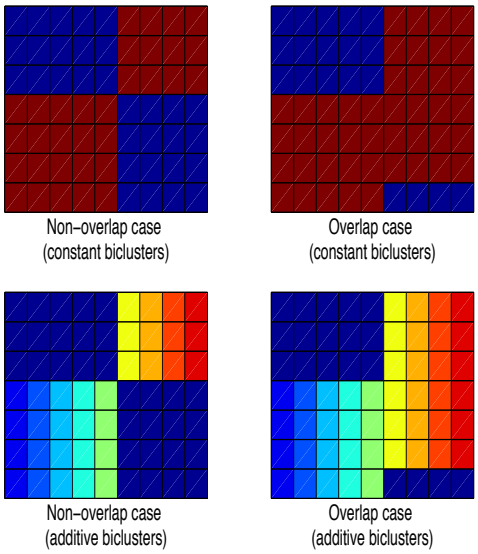

Figure 1. Bicluster Structures in Synthetic Datasets

datasets are kept small and they are of size 100×100. Note that the size of the datasets does not restrict the generality of the experimental results as we are focusing on the inherent structures of the input matrix [23].

6.1.2 Results Analysis

We use the match score defined in [23] to assess the biclustering performance. Formally, If M_1, M_2 are two biclustering sets, the match score in attribute dimension of M_1 with respect to M_2 is:

$$S(M_1, M_2) = \frac{1}{|M_1|} \sum_{(G_1, C_1) \in M_1} \max_{(G_2, C_2) \in M_2} \frac{|G_2 \bigcap G_1|}{|G_2 \bigcup G_1|} \quad (9)$$

The match score in sample dimension can be defined similarly. In general, the higher the score, the better the clustering performance. The thresholding BMF algorithm is used in our experiments since the datasets are generally sparse. We compare it with four other methods, BiMax [23], ISA [11, 10], SAMBA [26],and Binary Non-Orthogonal Matrix Decomposition [19](BND for short). Note that BND is based on heuristics while BMF is based on non-linear programming. In addition, BND is sensitive to initialization of the iterative process. The first three algorithms have been reported to be the best among the six biclustering methods [23]. ISA and BiMax are implemented by the software BicAT developed by [23], SAMBA is implemented by EXPANDER [25], and BND is provided by PROXIMUS [19].

Figure 2 and Figure 3 present the results on synthetic datasets. We use match score $G(M_{opt}, M_{comp})$ as the standard to assess the performance, where M_{opt} is the implanted biclustering structure and M_{comp} is the computed biclustering structure. From Figure 2 and Figure 3, we observe that: i) the thresholding BMF is almost noise-independent and overlap degree-independent; ii) the thresholding BMF is always the best among the four methods and can nearly identify all the bicluster structures. The main reason is the ability of BMF to correctly discretize original matrix. This is one of the key characteristics of BMF and is very important for identifying the exact bicluster structures. The results also show that, unlike the other greedy search strategy-based algorithms, BMF is more likely to find the global optima.

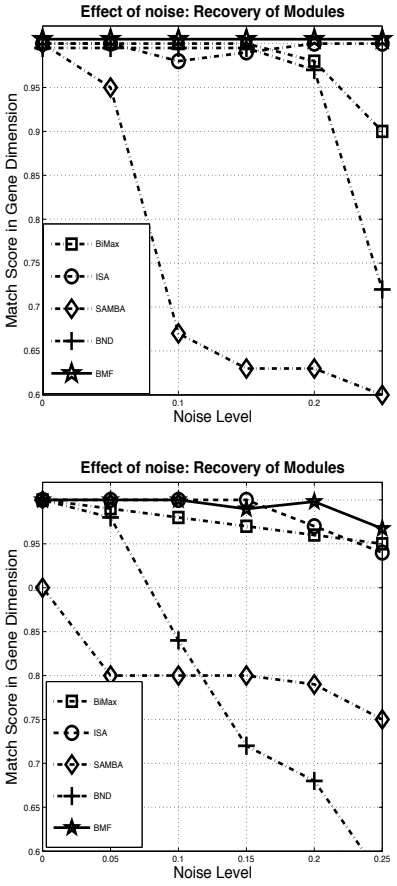

Figure 2. Performance on non-overlap case. BinaryNMF represents the thresholding BMF. The top figure shows the results for constant biclusters and the bottom figures shows the results for additive biclusters.

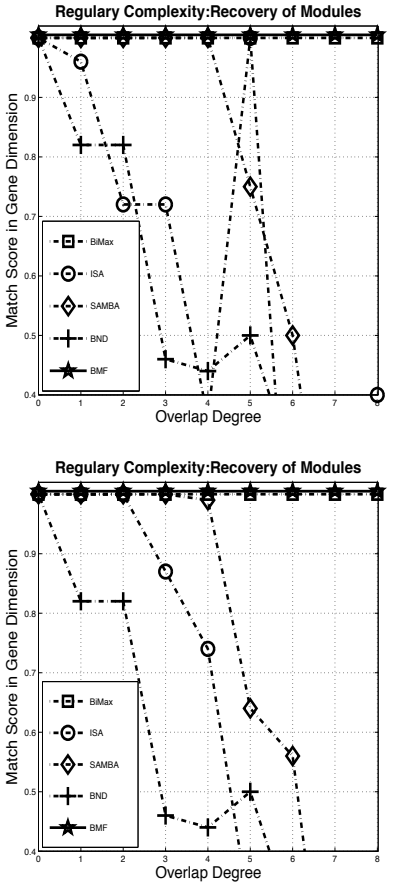

Figure 3. Performance on overlap case. BinaryNMF represents the thresholding BMF. The top figure shows the results for constant biclusters and the bottom figures shows the results for additive biclusters.

6.2. Gene Expression Datasets

In this section, we perform experiments to identify the bicluster structures on real world gene expression datasets.

6.2.1 Datasets Description

Three gene expression datasets: AML/ALL data [2], lung cancer data [9], and Central Nervous System tumor data [2], are used in our experiments. The **ALL/AML**dataset includes two types of human tumor-acute myelogenous leukemia (AML, 11 samples) and acute lymphoblastic leukemia (ALL, 27 samples). Also ALL can be divided into two subtypes-ALL-T(8 samples) and ALL-B(19 samples). The **Central Nervous System (CNS)** dataset consists of 34 samples: 10 classic medulloblastomas, 10 malignant, gliomas, 10 rhabdoids and 4 normals. The **Lung Cancer (LC)** dataset is composed of 32 samples including malignant pleural mesothelioma (MPM, 16 samples) and adenocarcinoma (ADCA, 16 samples). The datasets and their characteristics are summarized in Table 3.

In our experiment, the genes are ranked according to their coefficient of variation (i.e., standard deviation divided by the mean) and the top genes are selected. The gene expression data can be represented as a matrix X of $n \times m$, the i-th row of which represents the i-th gene's expression level across the m different samples. X is first discretized into binary matrix using the method described in [23]. In the resulting matrix, the element a_{ij} denotes whether the gene i is active in the sample j or not.

Datasets	# samples	# genes	# class
ALL/AML	38	5000	2
ALL/AML	38	5000	3
Lung Cancer	32	5000	2
CNS	34	5597	4

Table 3. Description of Gene Expression Datasets

6.2.2 Result Analysis

Table 4 shows the results on gene expression datasets. We compare our thresholding BMF method with SAMBA. Note that the results of BiMax and ISA are not included since they are either time-consuming for large datasets or do not yield any robust biclustering results. As one can see, the match scores of the thresholding BMF is consistently higher than those of SAMBA. Although the detailed bicluster structures are unknown, the results indicate that BMF is a promising model for biclustering. In addition, SAMBA generates many bicluster structures among which some are obviously meaningless as the match score in sample dimension based on these structures are very low.

	BMF	Samba
AML/ALL,(K=2)	96.3%	75.7%
AML/ALL,(K=3)	91.2%	75.7%
Lung Cancer	96.9%	81.8%
CNS	82.5%	66.1%

Table 4. Comparison on Gene Expression Datasets

6.3. Document Clustering

In this section, experiments are conducted on document datasets to evaluate the performance of BMF methods. In our experiments, documents are represented using the binary vector-space model where each document is a binary vector in the term space. Since the document datasets are usually sparse, so the thresholding method is used in experiments. We also compare our BMF algorithm with K-means and standard NMF algorithms.

6.3.1 Datasets Description

We use a variety of datasets, most of which are frequently used in the data mining research. Table 5 summarizes the characteristics of the document datasets. More detailed description of these datasets can be found in [17, 16]. To pre-process the datasets, we remove the stop words using a standard stop list, all HTML tags are skipped and all header fields except subject and organization of the posted articles are ignored. In all our experiments, we first select the top 1000 words by mutual information with class labels. The feature selection is done with the rainbow package [20].

Datasets	# documents	# class
CSTR	476	4
WebKB4	4199	4
Reuters	2,900	10
WebACE	2,340	20
Log	1367	9

Table 5. Document Datasets Descriptions.

6.3.2 Results Analysis

The above document datasets are standard labeled corpora. We view the labels of the datasets as the objective knowledge on the structure of the datasets. We use accuracy as the clustering performance measure. Accuracy discovers the one-to-one relationship between clusters and classes and measures the extent to which each cluster contained data points from the corresponding class [16]. The experimental results are shown in Table 6.

Datasets/Methods	K-Means	NMF	BMF
CSTR	0.43	0.57	0.56
WebKB4	0.38	0.44	0.42
Reuters	0.44	0.50	0.51
WebACE	0.40	0.48	0.47
Log	0.69	0.78	0.79

Table 6. Clustering Accuracy. Each entry is the clustering accuracy of the column method on the corresponding row dataset. The results obtained by averaging 10 trials.

From the experimental comparisons, we observe that: (i) Both NMF and BMF algorithms outperform the K-means clustering algorithm. As discussed in [4], NMF is equivalent to soft K-means and the soft relaxation improves clustering performance. (ii) The accuracy results of NMF and BMF are really close and their differences are small. On CSTR, WebKB4 and WebACE datasets, NMF is slightly better than BMF; while BMF is slightly better than NMF on Reuters and Log datasets. (iii) In general, BMF is a restricted form of NMF. Earlier studies [3, 7] have discussed the biclustering aspect of NMF. But the key difficulty is that one can not identify the binary relationship exactly as the resulting matrices W and H are not binary. However, BMF

can explicitly identify the co-association relationships between the documents and terms since W and H are binary. Hence, BMF is a competitive option for binary data clustering, especially when interpretability is viewed as a goal of the data analysis. (iv) Since W and H are binary, BMF offers a framework for simultaneously clustering the documents and terms. The framework is able to perform implicit feature selection and provide adaptive metrics for document clustering. Both of these properties are preferable for clustering in high-dimensional data.

7. Conclusion

In this paper, we extend the standard NMF to BMF: given a binary input matrix X, we want to factorize X into two binary matrices W, H satisfying $X \approx WH$. BMF preserves the most important integer property of X. Two factorization methods: penalty function and thresholding, are proposed and studied. We also prove the boundedness theorem to eliminate the bias of factorization and make our proposed methods more robust. Our study suggests that the penalty function method works better when the input binary matrix X is dense while the thresholding method excels otherwise. Experimental results show the usefulness and competitiveness of BMF.

Acknowledgments: This work is partially supported by the National Natural Science Foundation of China under grant No.10631070, and the Ministry of Science and Technology, China, under grant No.2006CB503905. Tao Li is partially supported by a IBM Faculty Research Award, NSF CAREER Award IIS-0546280 and NIH/NIGMS S06 GM008205. Chris Ding is supported in part by a University of Texas STARS Award.

References

[1] C. C. Aggarwal, J. L. Wolf, P. S. Yu, C. Procopiuc, and J. S. Park. Fast algorithms for projected clustering. In *SIGMOD*, pages 61–72, 1999.

[2] J.-P. Brunet, P. Tamayo, T. R. Golub, and J. P. Mesirov. Metagenes and molecular pattern discovery using matrix factorization. *Proceedings of the National Academy of Science*, 101(12):4164–4169, 2004.

[3] P. Carmona-Saez, R. D. Pascual-Marqui, F. Tirado, J. M. Carazo, and A. Pascual-Montano. Biclustering of gene expression data by non-smooth non-negative matrix factorization. *BMC Bioinformatics*, 7:78, 2006.

[4] C. Ding, X. He, and H. Simon. On the equivalence of nonnegative matrix factorization and spectral clustering. *Proc. SIAM Data Mining Conf*, 2005.

[5] C. Ding, T. Li, W. Peng, and H. Park. Orthogonal nonnegative matrix tri-factorizations for clustering. In *SIGKDD*, pages 126–135, 2006.

[6] C. Eckart and G. Young. The approximation of one matrix by another of lower rank. *Psychometrika*, 1:183–187, 1936.

[7] Y. Gao and G. Church. Improving molecular cancer class discovery through sparse non-negative matrix factorization. *Bioinformatics*, 21(21):3970–3975, 2005.

[8] G. Golub and C. Van Loan. *Matrix Computations, 3rd edition*. Johns Hopkins, Baltimore, 1996.

[9] G. J. Gordon, R. V. Jensen, L.-L. Hsiao, S. R. Gullans, J. E. Blumenstock, S. Ramaswamy, W. G. Richards, D. J. Sugarbaker, and R. Bueno. Translation of microarray data into clinically relevant cancer diagnostic tests using gene expression ratios in lung cancer and mesothelioma. *Cancer Research*, 62:4963–4967, 2002.

[10] J. Ihmels, S. Bergmann, and N. Barkai. Defining transcription modules using large-scale gene expression data. *Bioinformatics*, 20(13):1993–2003, 2004.

[11] J. Ihmels, G. Friedlander, S. Bergmann, O. Sarig, Y. Ziv, and N. Barkai. Revealing modular organization in the yeast transcriptional network. *Nature Genet*, 31:370–377, 2002.

[12] F. D. la Torre and T. Kanade. Discriminative cluster analysis. In *ICML*, 2006.

[13] D. Lee and H. S. Seung. Learning the parts of objects by non-negative matrix factorization. *Nature*, 401:788–791, 1999.

[14] D. Lee and H. S. Seung. Algorithms for non-negatvie matrix factorization. In *NIPS*, 2001.

[15] T. Li. A general model for clustering binary data. In *SIGKDD*, pages 188–197, 2005.

[16] T. Li and C. Ding. The relationships among various nonnegative matrix factorization methods for clustering. In *ICDM*, pages 362–371, 2006.

[17] T. Li, S. Ma, and M. Ogihara. Document clustering via adaptive subspace iteration. In *SIGIR*, pages 218–225, 2004.

[18] B. Long, Z. Zhang, and P. Yu. Co-clustering by block value decomposition. In *SIGKDD*, pages 635–640, 2005.

[19] A. G. M Koyuturk and N. Ramakrishnan. Non-orthogonal decomposition of binary matrices for bounded-error data compression and analysis. *ACM Transactions on Mathematical Software*, 32(1):33–69, March 2006.

[20] A. McCallum. Bow: A toolkit for statistical language modeling, text retrieval, classification and clustering. http://www.cs.cmu.edu/ mccallum/bow, 1996.

[21] P. Miettinen, T. Mielikäinen, A. Gionis, G. Das, and H. Mannila. The discrete basis problem. In *PKDD*, pages 335–346, 2006.

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

[23] A. Prelic;, S. Bleuler, P. Zimmermann, A. Wille, P. Buhlmann, W. Gruissem, L. Hennig, L. Thiele, and E. Zitzler. A systematic comparison and evaluation of biclustering methods for gene expression data. *Bioinformatics*, 22(9):1122–1129, 2006.

[24] F. Sha, L. Saul, and D. Lee. Multiplicative updates for nonnegative quadratic programming in support vector machines. In *NIPS*, pages 1041–1048. 2003.

[25] R. Sharan, A. Maron-Katz, and R. Shamir. Click and expander: a system for clustering and visualizing gene expression data. *Bioinformatics*, 19(14):1787–1799, 2003.

[26] A. Tanay, R. Sharan, and R. Shamir. Discovering statistically significant biclusters in gene expression data. *Bioinformatics*, 18(90001):S136–S144, 2002.

[27] D. Zeimpekis and E. Gallopoulos. Clsi: A flexible approximation scheme from clustered term-document matrices. *Proc. SIAM Data Mining Conf*, pages 631–635, 2005.