Deep Graph semi-NMF algorithm and its convergence

Haonan Huang[a], Zuyuan Yang[a,*], Zhenni Li[a,b], Weijun Sun[a]

[a]School of Automation, Guangdong Key Laboratory of IoT Information Technology, Guangdong University of Technology, Guangzhou, 510006, China
[b]Guangdong-HongKong-Macao Joint Laboratory for Smart Discrete Manufacturing, Guangzhou 510006, China

Abstract

Non-negative matrix factorization (NMF) is a very useful method for learning data representation in lower-dimensional space. To further obtain the hidden information of the high-dimensional data, a novel deep graph based semi-NMF (DGsnMF) method is proposed in this paper. In our model, the deep factorization scheme is employed to explore the inner information of the data, and the graph constraint is used to keep the geometrical structures of the data when the dimension is reduced. Furthermore, a forward-backward splitting approach is developed to deal with the involved nonconvex optimization problem, which exists widely in deep factorization. After that, a rigorous convergence analysis is attached to the proposed algorithm and it is proved to converge to a critical point. The image clustering experiments show that the proposed method provides performance superior to that of comparable approaches.

Keywords: non-negative matrix factorization, deep matrix factorization, graph regularization, nonconvex optimization

1. Introduction

In recent years, non-negative matrix factorization (NMF)[1] has become popular data representation method because it can provide parts-of-whole explanations and has good clustering performance. As a useful dimensionality reduction tool, NMF has been used to deal with a number of problems including, but not limited to, image clustering, data mining, document clustering, and others. Given a non-negative data matrix $\mathbf{X}$, NMF focuses on finding two lower-dimensional matrices $\mathbf{W}$ and $\mathbf{H}$ with non-negative constraints, and their product has a good approximation of the original matrix, such that $\mathbf{X} \approx \mathbf{WH}$ with $\mathbf{W} \geq 0$ and $\mathbf{H} \geq 0$.

Many NMF variants are proposed to achieve better decomposition quality and improve the methods' data representation ability. Cai et al. [2] proposed a graph regularized non-negative matrix factorization (GNMF) by encoding the geometrical information of original data into a $P-$nearest neighbor graph. This approach can find compact representations and usually lead to better clustering results. Ding et al. [3] proposed Semi-NMF which only restricted the second submatrix $\mathbf{H}$ to containing nonnegative entries. This approach is equivalent to a soft version of k-means clustering and extends the applicable range of NMF methods.

The above NMF and its variants are all single layer structures. However, many real world datasets are complex and contain hidden hierarchical information which is hard to be extracted by using single layer methods. Very recently, as the development and success of deep learning methods in various areas [4], deep non-negative matrix factorization (DMF) techniques [5] have been attracted lots of interest, because the data representations learned through multiple components will enable better understanding of data. Basic DMF model focuses on find l weighted matrices and a new lower-dimensional representation $\mathbf{H}_l$, such that $\mathbf{X} \approx \mathbf{W}_1\mathbf{W}_2\cdots\mathbf{W}_l\mathbf{H}_l$.

According to the existing work, DMF has been proved to be very good at discovering intricate structures in high-dimensional data and is therefore applicable to solve many machine learning tasks. Trigeorgis et al. [6] extended shallow Semi-NMF into deep Semi-NMF structure and also considered nonlinear activation on factor matrices, which is able to learn better low dimensional attribute representations for clustering. Yu et al. [7] presented an unsupervised deep non-smooth NMF model (Deep nsNMF) by using Nesterov's accelerated gradient descent algorithm. Deep nsNMF can learn part-based and hierarchical representations simultaneously. Subsequently, Li et al. [8] integrated nonlinear activation function and extreme learning machine into deep nsNMF, and built a supervised deep nsNMF network for synthetic aperture radar image change detection. Feng et al. [9] presented a sparsity-constrained deep non-negative matrix factorization by enforcing $L_{1/2}$ constraint [10] and total variation for hyperspectral unmixing. It is worth noting that the deep matrix factorization methods have outstanding performance in many fields, including but not limited to multi-view learning [11], remote sensing image processing [12], community detection [13], and so on. The DMF with respect to a set of variables is more complex than the traditional NMF, leading to the convergence property difficult to analyze. Hence, how to prove that the whole sequence obtained by the algorithm converges to a critical point of the objective function is an open problem in the DMF problem.

In this paper, motivated by the advantages of the graph regu-

Email addresses: libertyhhn@foxmail.com (Haonan Huang), yangzuyuan@gdut.edu.cn (Zuyuan Yang), lizhenni2012@gmail.com (Zhenni Li), gdutswj@gdut.edu.cn (Weijun Sun)

[1] * Corresponding author.

larizer theory in matrix factorization model and deep learning, we propose a novel unsupervised DMF model where semi non-negative constraint and graph learning for data representation are considered, which named Deep Graph semi non-negative Matrix Factorization (DGsnMF). Semi non-negative constraint can effectively extend the applicability of NMF in cases where the data matrix is not strictly non-negative. We construct the neighbors graph term to make full use of the geometrical information and local invariance of the original data. Motivated by recent work on forward-backward splitting (FBS) algorithm [14, 15], we use the FBS framework to deal with the nonconvex, nonsmooth constraints of the proposed model and prove the convergence properties of the proposed algorithms. To empirically verify the effectiveness of our method, image clustering experiments are conducted on four widely used real-world datasets. The achieved outperforming results show the superiority of this method by comparing with several representative methods.

The main contributions of this work are summarized as follows.

- We present a novel unsupervised DMF model with manifold learning which can learn a compact hidden layer representation based on unknown and diverse attributes of the complex high-dimensional input data. With the graph regularization, our method can respect the data intrinsic geometric structure and have more discriminating power than other methods.
- To solve the DGsnMF with nonconvex optimization problem, we develop an algorithm called forward-backward splitting (FBS) and provide the complexity analysis. Our proposed algorithm is easy to implement and computationally efficient in practice.
- By theoretical and experimental analysis, we provide the convergence proof that the whole sequence of iterates is convergent and converges to a critical point. To the best of our knowledge, this is the first work to prove the theoretical convergence properties for the DMF problem with graph regularization.

The rest of this paper is organized as follows. In Section 2, we introduce the classical single layer NMF, Semi-NMF and the deep structure NMF variants. In Section 3, we formulate our mathematical model, develop an efficient algorithm of DGsnMF and prove its convergence. In Section 4, we compare our methods with other related works, and demonstrate the effectiveness and superiority of our method. Finally, concluding remarks and future work are drawn in Section 5.

2. Methods

Before introducing the details of our method, we will briefly review previous research in single-layer matrix factorization structure and multi-layers matrix factorization structure, respectively.

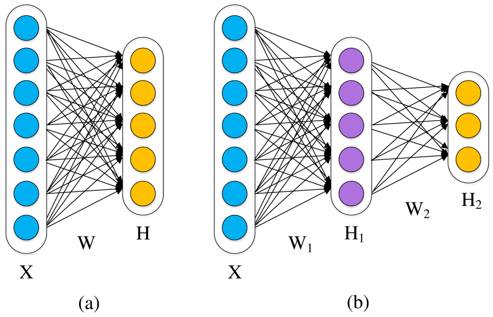


Figure 1: (a) The architecture of single layer matrix factorization. (b) The architecture of two layers Deep Semi-NMF

2.1. Single-Layer Matrix Factorization and Multi-Layers Variant

The classical single layer matrix factorization structure as illustrated in Figure 1 (a). Lee and Seung [1] proposed non-negative matrix factorization, which decomposes the given non-negative matrix $\mathbf{X} \in \mathbb{R}^{M\times N}$ into the product of two non-negative matrices weighted $\mathbf{W} \in \mathbb{R}^{M\times r}$ and features $\mathbf{H} \in \mathbb{R}^{r\times N}$, $r \leq \min\{M, N\}$. To extend the applicability of NMF in cases where the input data is not strictly non-negative and motivated by a clustering perspective, Ding et al. [3] proposed Semi-NMF which relaxes the non-negativity constrains of NMF. Semi-NMF only restricts feature matrix $\mathbf{H}$ to be non-negative while placing no restriction on weighted matrix $\mathbf{W}$. Therefore, the objective function of Semi-NMF is formulated as follows:

$$\min_{\mathbf{W},\mathbf{H}} \frac{1}{2}\|\mathbf{X} - \mathbf{W}\mathbf{H}\|_F^2 \quad \text{s.t.} \quad \mathbf{H} \geq 0. \tag{1}$$

In order to explore the complex hierarchical and structural information of input data, motivated by the idea of the deep representation learning [19], Trigeorgis et al. [6] presented an unsupervised multi-layer matrix factorization structures, which called Deep Semi-NMF. The two layers Deep Semi-NMF architecture as illustrated in Figure 1 (b). Deep Semi-NMF can learn hidden representations well and allow themselves to a clearly interpretation of clustering according to different unknown attributes of given data. However, Deep Semi-NMF ignores the geometric structural relations in the low-dimension representation.

2.2. Deep Graph Regularization Semi-nonnegative Matrix Factorization(DGsnMF)

Input a data matrix $\mathbf{X} \in \mathbb{R}^{M\times N}$, DGsnMF aims to find l weight matrices $\mathbf{W}_1 \in \mathbb{R}^{M\times r_1}, \mathbf{W}_2 \in \mathbb{R}^{r_1\times r_2}, ..., \mathbf{W}_l \in \mathbb{R}^{r_{l-1}\times r_l}$ and the lower-dimensional compact representation matrix $\mathbf{H}_l \in \mathbb{R}^{r_l\times N}$ whose product can approximate the original data $\mathbf{X}$:

$$\mathbf{X} \approx \mathbf{W}_1\mathbf{W}_2 \cdots \mathbf{W}_l\mathbf{H}_l$$

It should be noted that DGsnMF restrict only the features matrix $\mathbf{H}_l$ with non-negativity constrains and when the layer $i < l$, $\mathbf{H}_i = \mathbf{W}_{i+1}\mathbf{H}_{i+1} = \mathbf{W}_{i+1}\mathbf{W}_{i+2}...\mathbf{W}_l\mathbf{H}_l$.

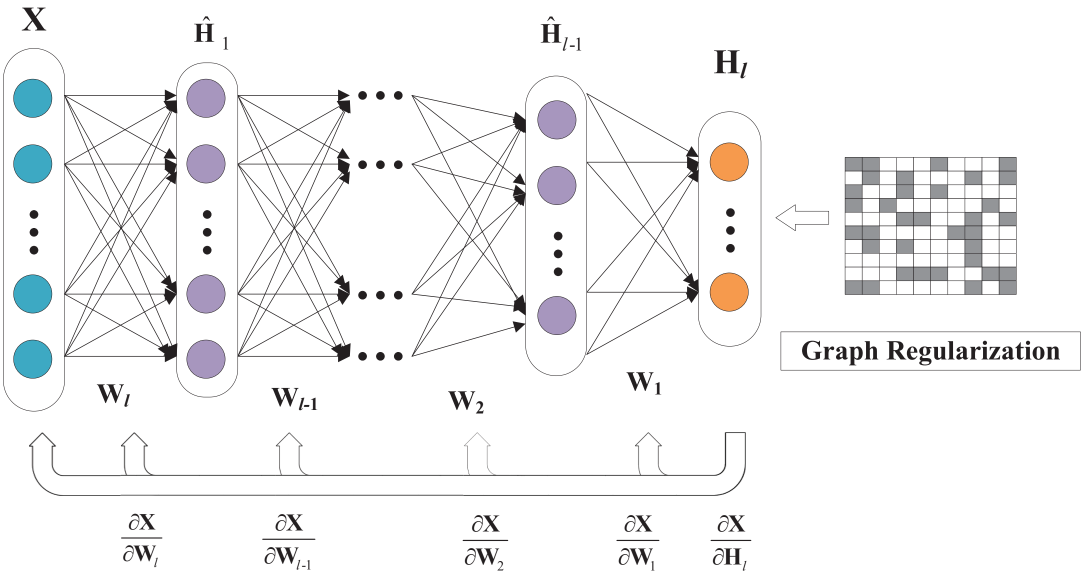

X
$\hat{\mathbf{H}}_1$
$\hat{\mathbf{H}}_{l\text{-}1}$
$\mathbf{H}_l$
Graph Regularization
$\mathbf{W}_l$
$\mathbf{W}_{l\text{-}1}$
$\mathbf{W}_2$
$\mathbf{W}_1$
$\frac{\partial \mathbf{X}}{\partial \mathbf{W}_l}$
$\frac{\partial \mathbf{X}}{\partial \mathbf{W}_{l\text{-}1}}$
$\frac{\partial \mathbf{X}}{\partial \mathbf{W}_2}$
$\frac{\partial \mathbf{X}}{\partial \mathbf{W}_1}$
$\frac{\partial \mathbf{X}}{\partial \mathbf{H}_l}$

Figure 2: Framework of the proposed DGsnMF method with l layers

In order to model the intrinsic geometric structure of original data, we construct a nearest neighbor graph on data points and define graph weight matrix $\mathbf{S}$ as follows:

$$\mathbf{S}_{jq} = \begin{cases} 1 & \text{if } y_q = y_j \\ 0 & \text{otherwise} \end{cases} \tag{2}$$

Graph Laplacian matrix $\mathbf{L}$ is defined as $\mathbf{L} = \mathbf{D} - \mathbf{S}$, where $\mathbf{D}$ is a diagonal matrix whose entries are row(or column, since $\mathbf{S}$ is symmetric) sums of $\mathbf{S}$, $\mathbf{D}_{jj} = \sum_q \mathbf{S}_{jq}$. Therefore, we can use the regularization term $\mathcal{R}$ to measure the smoothness of the lower-dimensional representation, as follows:

$$\begin{aligned} \mathcal{R} &= \frac{1}{2}\sum_{j,q=1}^{N} \left\|\mathbf{h}_j - \mathbf{h}_q\right\|^2 \mathbf{S}_{jq} \\ &= \sum_{j=1}^{N} \mathbf{h}_j^{\mathrm{T}}\mathbf{h}_j\mathbf{D}_{jj} - \sum_{j,q=1}^{N} \mathbf{h}_j^{\mathrm{T}}\mathbf{h}_l\mathbf{S}_{jq} \\ &= \mathrm{Tr}\left(\mathbf{H}^{\mathrm{T}}\mathbf{D}\mathbf{H}\right) - \mathrm{Tr}\left(\mathbf{H}^{\mathrm{T}}\mathbf{S}\mathbf{H}\right) = \mathrm{Tr}\left(\mathbf{H}^{\mathrm{T}}\mathbf{L}\mathbf{H}\right) \end{aligned} \tag{3}$$

where $\mathbf{h}_j$ is the low-dimensional representation for sample j and Tr(.) denotes the trace of a matrix.

Combining the above graph regularization term, we construct the objective function of DGsnMF as follows:

$$\min_{\mathbf{W}_1,\mathbf{W}_2,\ldots,\mathbf{W}_l,\mathbf{H}_l} \frac{1}{2}\|\mathbf{X} - \mathbf{W}_1\mathbf{W}_2\cdots\mathbf{W}_l\mathbf{H}_l\|_F^2 + \lambda\mathrm{Tr}(\mathbf{H}_l^{\mathrm{T}}\mathbf{L}\mathbf{H}_l) \qquad \text{s.t.} \quad \mathbf{H}_l \geq 0. \tag{4}$$

where λ denotes the graph regularization parameter and we normalize each weighted matrix $\mathbf{W}_i$ to prevent the scaling ambiguity. The framework of the DGsnMF as shown in Figure 2.

3. Algorithm

The DMF problems have a set of variables, its optimization is more complex than traditional single layer NMF. Multiplicative alternate optimization of $\mathbf{W}$ and $\mathbf{H}$ is proposed in [6], but there are calculation steps for calculating the square root and the pseudo-inverse, both steps require a huge memory and a large number of computations. The convergence properties of algorithm in DMF is difficult to analyze and cause the problem of redundant iterations. In this section, we developed a novel algorithm based on forward-backward splitting framework for the solution of our non-convex model.

3.1. Forward-Backward Splitting for DGsnMF

Firstly, we briefly review the Forward-Backward Splitting algorithm (FBS). The aim of the FBS framework is to solve the following model:

$$\arg\min_{\mathbf{x}} \phi(\mathbf{x}) = Q(\mathbf{x}) + I(\mathbf{x}), \tag{5}$$

where $Q(\mathbf{x})$ is a C^1 function[2], the gradient ∇Q is Lipschitz continuous and function I is proper and lower semi-continuous functions. We set L_Q denote the global Lipschitz constant of gradient ∇Q and assume there is sequence η^k which satisfies $0 < \frac{1}{\eta^k} \leq \frac{1}{L_Q}$. The gradient can be viewed as a proximal regularization [20] of the function Q at $\mathbf{x}^{k+1}$, and written equivalently as:

$$\mathbf{x}^{k+1} = \arg\min_{\mathbf{x}} \hat{Q}(\mathbf{x}) + I(\mathbf{x}) + \frac{\eta^k}{2}\|\mathbf{x} - \mathbf{x}^{k+1}\|_F^2 \tag{6}$$

[2] C^1 function: the first derivatives are continuous

where $\hat{Q}(\mathbf{x}) = Q(\mathbf{x}^k) + trace\langle \mathbf{x} - \mathbf{x}^k, \nabla_{\mathbf{x}} Q(\mathbf{x}^k)\rangle$ and the iteration (6) can be rewritten as,

$$\mathbf{x}^{k+1} = \arg\min_{\mathbf{x}} \frac{L_Q^k}{2} \left\| \mathbf{x} - \left(\mathbf{x}^k - \frac{1}{\eta^k} \nabla_{\mathbf{x}} Q(\mathbf{x}^k) \right) \right\|_F^2 + I(\mathbf{x}). \tag{7}$$

Using the forward-backward splitting algorithm, the formula (7) can be rewritten as

$$\mathbf{x}^{k+1} = P_I \left(\mathbf{x}^k - \frac{1}{\eta^k} \nabla_{\mathbf{x}} Q(\mathbf{x}^k) \right). \tag{8}$$

where P refers to the proximal mapping.

Secondly, we apply the Forward-Backward Splitting framework to the DGsnMF problem and (4) can be rewritten as the minimization problem,

$$\begin{aligned} \min_{\mathbf{W}_1,...,\mathbf{W}_l,\mathbf{H}_l} \phi(\mathbf{W}_1, ..., \mathbf{W}_l, \mathbf{H}_l) = Q(\mathbf{W}_1, ..., \mathbf{W}_l, \mathbf{H}_l) + \\ I(\mathbf{W}_1, ..., \mathbf{W}_l, \mathbf{H}_l), \end{aligned} \tag{9}$$

where

$$Q(\mathbf{W}_1, ..., \mathbf{W}_l, \mathbf{H}_l) = \frac{1}{2} \|\mathbf{X} - \mathbf{W}_1 \cdots \mathbf{W}_l \mathbf{H}_l\|_F^2 + \lambda \mathrm{Tr}(\mathbf{H}_l^{\mathrm{T}} \mathbf{L} \mathbf{H}_l), \tag{10}$$

and

$$I(\mathbf{W}_1, ..., \mathbf{W}_l, \mathbf{H}_l) = F(\mathbf{W}_i) + G(\mathbf{H}_l), \ i = 1, ..., l \tag{11}$$

where F or G is the indicator function δ of a nonempty closed set S. For all $x \in \mathbb{R}$, $\delta(x)$ is defined as follows,

$$\delta_S(x) = \begin{cases} 0, & if \ x \in S \\ \infty, & otherwise. \end{cases} \tag{12}$$

In this problem, $F(\mathbf{W}_i) = \delta_{S_F}(\mathbf{W}_i)$ and set S_F is a unit-norm sphere used to constrain the lengths of the weighted matrices. $G(\mathbf{H}_l) = \delta_{S_G}(\mathbf{H}_l)$ and set S_G is used to maintain the non-negative structure of the feature matrices.

Update $\mathbf{W}_i$: We fix feature matrix $\mathbf{H}_l$ and the other weighted matrix $\mathbf{W}_j (j \in 1, ..., l$ and $j \neq i)$ and minimize the cost function with respect to $\mathbf{W}_i$,

$$\min_{\mathbf{W}_i} \phi(\mathbf{W}_i) = Q(\mathbf{W}_1, .\mathbf{W}_i.., \mathbf{W}_l, \mathbf{H}_l) + F(\mathbf{W}_i), \ i = 1, ..., l \tag{13}$$

Update $\mathbf{H}_l$: We minimize the cost function with respect to $\mathbf{H}_l$ through fixing all weighted matrices $\mathbf{W}_i$,

$$\min_{\mathbf{H}_l} \phi(\mathbf{H}_l) = Q(\mathbf{W}_1, ..., \mathbf{W}_l, \mathbf{H}_l) + G(\mathbf{H}_l), \tag{14}$$

Applying the FBS to the problem (13) and (14), the iteration of $(\mathbf{W}_i^{k+1}, \mathbf{H}_l^{k+1})$ via as follows,

$$\mathbf{W}_i^{k+1} = \arg\min_{\mathbf{W}_i} \hat{Q}(\mathbf{W}_i) + F(\mathbf{W}_i) + \frac{\mu_i^k}{2} \|\mathbf{W}_i - \mathbf{W}_i^{k+1}\|_F^2 \tag{15}$$

$$\mathbf{H}_l^{k+1} = \arg\min_{\mathbf{H}_l} \hat{Q}(\mathbf{H}_l) + G(\mathbf{H}_l) + \frac{\nu^k}{2} \|\mathbf{H}_l - \mathbf{H}_l^{k+1}\|_F^2 \tag{16}$$

where $\hat{Q}(\mathbf{W}_i) = Q(\mathbf{W}_1^{k+1}, ..., \mathbf{W}_i^k, ..., \mathbf{W}_l^k, \mathbf{H}_l^k) + trace\langle \mathbf{W}_i - \mathbf{W}_i^{k+1}, \nabla_{\mathbf{W}_i} Q(\mathbf{W}_1^{k+1}, ..., \mathbf{W}_i^k, ..., \mathbf{W}_l^k, \mathbf{H}_l^k)\rangle$ and $\hat{Q}(\mathbf{H}_l) = Q(\mathbf{W}_1^{k+1}, ..., \mathbf{W}_i^{k+1}, ..., \mathbf{W}_l^{k+1}, \mathbf{H}_l^k) + trace\langle \mathbf{H}_l - \mathbf{H}_l^{k+1}, \nabla_{\mathbf{H}_l} Q(\mathbf{W}_1^{k+1}, ..., \mathbf{W}_i^{k+1}, ..., \mathbf{W}_l^{k+1}, \mathbf{H}_l^k)\rangle$, μ and ν are two positive suitable stepsizes. The iterations can be rewritten as,

$$\begin{aligned} \mathbf{W}_i^{k+1} = \arg\min_{\mathbf{W}_i} F(\mathbf{W}_i) + \\ \frac{\mu_i^k}{2} \left\| \mathbf{W}_i - \left(\mathbf{W}_i^k - \frac{1}{\mu_i^k} \nabla_{\mathbf{W}_i^k} Q(\mathbf{W}_1^{k+1}, ..., \mathbf{W}_i^k, ..., \mathbf{W}_l^k, \mathbf{H}_l^k) \right) \right\|_F^2 \end{aligned} \tag{17}$$

$$\begin{aligned} \mathbf{H}_l^{k+1} = \arg\min_{\mathbf{H}_l} G(\mathbf{H}_l) + \\ \frac{\nu^k}{2} \left\| \mathbf{H}_l - \left(\mathbf{H}_l^k - \frac{1}{\nu^k} \nabla_{\mathbf{H}_l^k} Q(\mathbf{W}_1^{k+1}, \mathbf{W}_2^{k+1}, ..., \mathbf{W}_l^{k+1}, \mathbf{H}_l^k) \right) \right\|_F^2 \end{aligned} \tag{18}$$

Since the indicator functions F and G are both separable, the update rule of (17) and (18) can be rewritten as follows,

$$\mathbf{W}_i^{k+1} = P_F \left(\mathbf{W}_i^k - \frac{1}{\mu_i^k} \nabla_{\mathbf{W}_i^k} Q(\mathbf{W}_1^{k+1}, ..., \mathbf{W}_i^k, ..., \mathbf{W}_l^k, \mathbf{H}_l^k) \right), \tag{19}$$

$$\mathbf{H}_l^{k+1} = P_G \left(\mathbf{H}_l^k - \frac{1}{\nu^k} \nabla_{\mathbf{H}_l^k} Q(\mathbf{W}_1^{k+1}, \mathbf{W}_2^{k+1}, ..., \mathbf{W}_l^{k+1}, \mathbf{H}_l^k) \right), \tag{20}$$

In this problem, the proximal mapping of indicator functions F and G are the projection to the closed sets. Therefore, given any matrix $\mathbf{V} \in \mathbb{R}^N$, functions $P_F(\mathbf{V})$ and $P_G(\mathbf{V})$ can be calculated as follows:

$$P_F(\mathbf{V}) = \frac{\mathbf{V}}{\|\mathbf{V}\|^2}, \quad if \ \mathbf{V} \neq 0 \tag{21}$$

$$P_G(\mathbf{V}) = \begin{cases} 0, & if \ \mathbf{V}_{ij} < 0 \\ \mathbf{V}_{ij}, & otherwise. \end{cases} \tag{22}$$

For the convenience of optimizing our algorithm, we introduce the auxiliary matrices $\hat{\mathbf{H}}_i$ and $\mathbf{Z}_i$

$$\hat{\mathbf{H}}_i = \{ \begin{smallmatrix} \mathbf{H}_l & ,if \ i=l \\ \mathbf{W}_{i+1} \mathbf{W}_{i+2} \cdots \mathbf{W}_l \mathbf{H}_l & ,otherwise \end{smallmatrix} \tag{23}$$

$$\mathbf{Z}_{i-1} = \{ \begin{smallmatrix} \mathbf{I} & ,if \ i=1 \\ \mathbf{W}_1 \mathbf{W}_2 \cdots \mathbf{W}_i & ,otherwise \end{smallmatrix} \tag{24}$$

formula (19) and (20) can be rewritten as

$$\mathbf{W}_i^{k+1} = P_F \left(\mathbf{W}_i^k - \frac{1}{\mu_i^k} \nabla_{\mathbf{W}_i^k} Q(\mathbf{Z}_{i-1}^{k+1}, \mathbf{W}_i^k, \hat{\mathbf{H}}_i^k) \right), \tag{25}$$

$$\mathbf{H}_l^{k+1} = P_G \left(\mathbf{H}_l^k - \frac{1}{\nu^k} \nabla_{\mathbf{H}_l^k} Q(\mathbf{Z}_l^{k+1}, \mathbf{H}_l^k) \right), \tag{26}$$

In order to use FBS frame to solve DGsnMF problem, we need the Lipschitz modulis of $\nabla_{\mathbf{W}_i^k} Q$ and $\nabla_{\mathbf{H}_l^k} Q$ to determine the step sizes μ_i and ν. By following the analysis and proof in [7, 21], we find that $\nabla_{\mathbf{W}_i^k} Q$ and $\nabla_{\mathbf{H}_l^k} Q$ are Lipschitz continuous and we can set the $\mu_i = \|\mathbf{Z}_{i-1}^{\mathrm{T}} \mathbf{Z}_{i-1}\|_2 \|\hat{\mathbf{H}}_i \hat{\mathbf{H}}_i^{\mathrm{T}}\|_2$ and $\nu = \|\mathbf{Z}_l^{\mathrm{T}} \mathbf{Z}_l\|_2 + 2\lambda \|\mathbf{L}\|_2$.

Given the $\mathbf{Z}_{i-1}$ and $\hat{\mathbf{H}}_i$, update $\mathbf{W}_i$, the objective function Q (10) can be rewritten as

$$\begin{aligned} Q(\mathbf{Z}_{i-1}, \mathbf{W}_i, \hat{\mathbf{H}}_i) &= \frac{1}{2}\|\mathbf{X} - \mathbf{Z}_{i-1}\mathbf{W}_i\hat{\mathbf{H}}_i\|_F^2 + \lambda \mathrm{Tr}(\mathbf{H}_l^{\mathrm{T}}\mathbf{L}\mathbf{H}_l) \\ &= \frac{1}{2}\mathrm{Tr}\left((\mathbf{X} - \mathbf{Z}_{\mathrm{i}-1}\mathbf{W}_{\mathrm{i}}\hat{\mathbf{H}})(\mathbf{X} - \mathbf{Z}_{\mathrm{i}-1}\mathbf{W}_{\mathrm{i}}\hat{\mathbf{H}})^{\mathrm{T}}\right) \\ &\quad + \lambda \mathrm{Tr}(\mathbf{H}_l^{\mathrm{T}}\mathbf{L}\mathbf{H}_l) \\ &= \frac{1}{2}\mathrm{Tr}(\mathbf{X}\mathbf{X}^{\mathrm{T}} - 2\mathbf{X}(\mathbf{Z}_{\mathrm{i}-1}\mathbf{W}_{\mathrm{i}}\hat{\mathbf{H}})^{\mathrm{T}} \\ &\quad + (\mathbf{Z}_{i-1}\mathbf{W}_i\hat{\mathbf{H}})(\mathbf{Z}_{i-1}\mathbf{W}_i\hat{\mathbf{H}})^{\mathrm{T}}) \\ &\quad + \lambda \mathrm{Tr}(\mathbf{H}_l^{\mathrm{T}}\mathbf{L}\mathbf{H}_l) \end{aligned} \tag{27}$$

By applying the matrix properties $\mathrm{Tr}(\mathbf{V}^{\mathrm{T}}) = \mathrm{Tr}(\mathbf{V})$ and $\mathrm{Tr}(\mathbf{U}\mathbf{V}) = \mathrm{Tr}(\mathbf{V}\mathbf{U})$, we can compute the derivative of the weight matrices $\mathbf{W}_i$,

$$\nabla_{\mathbf{W}_i} Q(\mathbf{Z}_{i-1}, \mathbf{W}_i, \hat{\mathbf{H}}_i) = (\mathbf{Z}_{i-1}^{\mathrm{T}}\mathbf{Z}_{i-1}\mathbf{W}_i\hat{\mathbf{H}}_i - \mathbf{Z}_{i-1}^{\mathrm{T}}\mathbf{X})\hat{\mathbf{H}}_i^{\mathrm{T}} \tag{28}$$

Fix $\mathbf{W}_1, \mathbf{W}_2, ..., \mathbf{W}_l$ and update $\mathbf{H}_l$. Given $\mathbf{Z}_l = \mathbf{W}_1\mathbf{W}_2 \cdots \mathbf{W}_l$, the objective function Q (10) can be rewritten as

$$\begin{aligned} Q(\mathbf{Z}_l, \mathbf{H}_l) &= \frac{1}{2}\|\mathbf{X} - \mathbf{Z}_l\mathbf{H}_l\|_F^2 + \lambda \mathrm{Tr}(\mathbf{H}_l^{\mathrm{T}}\mathbf{L}\mathbf{H}_l) \\ &= \frac{1}{2}\mathrm{Tr}\left((\mathbf{X} - \mathbf{Z}_l\mathbf{H}_l)(\mathbf{X} - \mathbf{Z}_l\mathbf{H}_l)^{\mathrm{T}}\right) + \lambda \mathrm{Tr}(\mathbf{H}_l^{\mathrm{T}}\mathbf{L}\mathbf{H}_l) \\ &= \frac{1}{2}\mathrm{Tr}\left(\mathbf{X}\mathbf{X}^{\mathrm{T}} - 2\mathbf{X}(\mathbf{Z}_l\mathbf{H}_l)^{\mathrm{T}} + (\mathbf{Z}_l\mathbf{H}_l)(\mathbf{Z}_l\mathbf{H}_l)^{\mathrm{T}}\right) \\ &\quad + \lambda \mathrm{Tr}(\mathbf{H}_l^{\mathrm{T}}\mathbf{L}\mathbf{H}_l) \end{aligned} \tag{29}$$

Similarly, we can calculate $\nabla_{\mathbf{H}_l} Q$ as follows:

$$\nabla_{\mathbf{H}_l} Q(\mathbf{Z}_l, \mathbf{H}_l) = \mathbf{Z}_l^{\mathrm{T}}(\mathbf{Z}_l\mathbf{H}_l - \mathbf{X}) + 2\lambda\mathbf{L}_l\mathbf{H}_l \tag{30}$$

With the above theoretical analysis, we summarize the algorithm using nonconvex Forward-Backward Splitting framework to solve DGsnMF in Algorithm 1.

```
Algorithm 1 The FBS Algorithm for DGsnMF.
Input: $\mathbf{X} \in \mathbb{R}^{M \times N}$,the number of layers $l$, the layers size
    $[r_1, r_2, ..., r_l]$ and hyperparameter $\lambda$
Randomly initialize $\mathbf{W}_i$ for each of the layers and $\mathbf{H}_l$
compute Laplacian Graph Matrix $\mathbf{L}$ and construct manifold
    regularization using Equations (2) (3), respectively
repeat   $k$ iterations
    for $i = 1, 2, ..., l$
        compute $\mathbf{W}_i^k$ using Equations (25)
    end for
    compute $\mathbf{H}_l^k$ using Equations (26)
until Stopping criterion is reached;
Output: Each layer weight matrix $\mathbf{W}_i$ and feature matrix $\mathbf{H}_l$.
```

3.2. Computational Complexity Analysis

According to Algorithm 1, the main time cost is spent on updating rules giving in Equations (25) and (26). The computational complexity of DGsnMF is of order $O(kl(MNr + Mr^2 + Nr^2))$, where k is the number of iterations to achieve convergence, l is the number of layers, and r is the maximal layer size out of all layers.

3.3. Convergence Analysis

Theorem 3.1. *The sequence $\{\boldsymbol{W}_1^k, ..., \boldsymbol{W}_l^k, \boldsymbol{H}_l^k\}$ generated by the Algorithm 1 have a finite length and converges to a critical point.*

Proof We use following Theorem 3.2 as the main tool for the proof. According to the following conditions, we check the sequence $\{\mathbf{J}^k\} = \{\mathbf{W}_1^k, \mathbf{W}_2^k, ..., \mathbf{W}_l^k, \mathbf{H}_l^k\}$ generated by the Algorithm 1 satisfies the conditions V1-V4. See Appendix A for details. ■

Theorem 3.2. *Let $\phi(\boldsymbol{J})$ be a proper and lower semi-continuous function which is bounded from below ($inf\{\phi\} > -\infty$) and the whole sequence $\{\boldsymbol{J}^k\}_{k \in \mathbb{N}}$ converges to a critical point of $\phi(\boldsymbol{J})$ if satisfies the following conditions[18]:*

V1: Sufficient decrease condition, there exists some constants $a > 0$ such that

$$\phi(\mathbf{J}^k) - \phi(\mathbf{J}^{k+1}) \geq a\|\mathbf{J}^{k+1} - \mathbf{J}^k\|_F^2, \qquad \forall k$$

V2: Relative error condition, there exists $\mathbf{N}^{k+1} \in \partial\phi(\mathbf{J}^k)$ and some constants $b > 0$ such that

$$\|\mathbf{N}^{k+1}\|_F^2 \leq b\|\mathbf{J}^{k+1} - \mathbf{J}^k\|_F^2, \qquad \forall k$$

V3: Continuity condition[20], exist $\{\mathbf{J}^{k_j}\}_{j \in \mathbf{N}}$ and $\tilde{\mathbf{J}}$

$$\mathbf{J}^{k_j} \to \tilde{\mathbf{J}} \text{ and } \phi(\mathbf{J}^{k_j}) \to \phi(\tilde{\mathbf{J}}), \qquad as\ j \to \infty$$

V4: $\phi(\mathbf{J})$ satisfies the Kurdyka-Łojasiewicz Property in its effective domain.

Proof The details of this proof is provided in Appendix A. ■

4. Experiments

In this section, we have tested the following real world datasets for clustering experiments:

1. **Yale**: Classical face database contains 165 grayscale images (32 × 32) of 15 individuals.There are 11 images per subject, one per different facial expression or configuration: center-light, w/glasses, happy, left-light, w/no glasses, normal, right-light, sad, sleepy, surprised, and wink.
2. **ORL**: This face database composed of 400 images of size 32×32 . There are 40 persons, 10 images per each person and all images were taken at different times, lighting and facial expressions.

3. **JAFFE**: The *Japanese Female Facial Expression* (JAFFE) [22] contains 213 facial grayscale images posed by 10 Japanese female models. All the images were resized to (32×32).
4. **PIE**: This is a freely face dataset, which comprises of 2856 grayscale 32×32 images of 68 subjects. Each person has 42 facial images under different illumination conditions.

The Yale, ORL and PIE datasets can be downloaded from the following link: http://featureselection.asu.edu/datasets.php

In order to evaluate the performance of our model, we compared our methods with the following unsupervised algorithms in experiments:

- The NMF method with multiplicative update rules proposed in [1].
- The Deep nsNMF model recently proposed in [7].
- The Semi-NMF proposed in [3] and its multi-layer version, Deep Semi-NMF model, which proposed in [6].

Similar to [6][7], all the deep architectures methods are comprised of two hidden layers and the details of layer size configuration is provided in Table 1. For example, for the PIE original data with 1024 features, following the parameters setting in [6], we experimented with DGsnMF that had a first hidden representation $\hat{\mathbf{H}}_1$ with 625 features, and a second representation $\mathbf{H}_2$ with a number of features a that ranged from 20 to 70. Therefore, the layer sizes are empirically searched in $1024 - 625 - a$.

Table 1: The layers configuration of all deep architectures methods in our experiments (the value of a from 20 to 70).

Dataset	Class	Size	Layers Configuration
Yale	15	165	$1024 - 165 - a$
ORL	40	400	$1024 - 400 - a$
JAFFE	10	213	$1024 - 200 - a$
PIE	68	2856	$1024 - 625 - a$

4.1. Implementation Details

To speed up the approximation of the factor matrices in the proposed model, we pre-train each of the layers to have an initial approximation of the factor matrices $\mathbf{W}_1, \mathbf{W}_2, ..., \mathbf{W}_l$ and $\mathbf{H}_l$. The effectiveness of pre-training scheme has been proven in [6][23], and it can greatly reduce the training time in experiments. To perform the pre-training, we first decompose the input matrix $\mathbf{X} \approx \mathbf{W}_1\mathbf{H}_1$ by minimizing $\|\mathbf{X} - \mathbf{W}_1\mathbf{H}_1\|_F^2 + \lambda \mathrm{Tr}(\mathbf{H}_1^{\mathrm{T}}\mathbf{L}\mathbf{H}_1)$, where $\mathbf{W}_1 \in \mathbb{R}^{M \times r_1}$ and $\mathbf{H}_1 \in \mathbb{R}_+^{r_1 \times N}$. Then, we decompose the matrix $\mathbf{H}_1$ as $\mathbf{H}_1 \approx \mathbf{W}_2\mathbf{H}_2$, and continue to do so until all of the layers have been pre-trained. When the convergence value is small enough, we use the convergence rule $Q_i - Q_{i+1} \leq 10^{-5} max(Q_i, 1)$ to stop the iterative process.

In order to assess the feature matrix $\mathbf{H}_l$ learned by DGsnMF, we use k-means clustering algorithm as in [2]. The Clustering Accuracy(ACC) and Normalized Mutual Information(NMI) are used as evaluation metrics to measure the clustering performance. These two metrics are detailed in [24]. It should be noted that both larger NMI and ACC indicate the better clustering performance.

We use GPU programming mode in MATLAB to speed up the calculation of our algorithm. All experiments are conducted on a server with two 3.10GHz Intel Xeon CPU E5-2687W and 256GB main memory running Ubuntu 16.04 (64-bit) with MATLAB R2016b.

4.2. Convergence and parameter analysis

We have proven that the updating algorithm of the proposed model are convergent, as described in Section 3.3. Following the parameters setting introduced in Table 1, we record the value of our DGsnMF problem as described in formula (9) with different top layer sizes. Figure 3 shows the performance of convergence value with iteration numbers in Yale, ORL, JAFFE and PIE, respectively. These experiments demonstrate a number of interesting points.

1. We observe that the convergence value decreases steadily and gradually reaches the convergence after a certain number of iterations, demonstrating that the proposed FBS optimization scheme is effective and converges quickly.
2. The convergence speed of the algorithm is also affected by the layer sizes. With the increase of the top layer sizes, the initial target value increases, and more iterations are needed to reduce to the same convergence value.
3. The convergence rate of our algorithm is different when testing different datasets, e.g., it has faster performance on JAFFE and needs the most iterations on PIE to reach the convergence state. This indicates that the convergence performance of our algorithm will be affected by different distributed data.

In DGsnMF, the graph regularization parameters λ is used to adjust the contribution of the graph constrains. Figure 4 and 5 show the NMI and ACC results of DGsnMF model with different hyper-parameters λ on four datasets, respectively. In these experiments, we fix the layer sizes on different datasets and the range of λ is set to $\{0.00001, 0.0001, 0.001, 0.01, 0.1, 1, 10\}$. From figure 4 and 5, we can see that the parameter λ obviously affects the experimental results in different datasets, which indicates that λ plays an important role in DGsnMF.

4.3. Clustering performance

Similar to [6], we evaluate the clustering performance in terms of ACC and NMI, with respect to different top layer sizes from 20 to 70. From our parameter analysis results as described in Section 4.2 and Figure 4-5, we choose the best λ parameter for each dataset (Yale-0.01, ORL-0.0001, JAFFE-0.001, PIE-0.01). Through using K-means algorithm to cluster the top layer low-dimensional data representations learned by different models, we evaluate the clustering results of five models on four databases are shown in Tables 2-5. We underline the best performance of each method and mark the best clustering result under the same top layer size in bold. From Tables 2-5, it can be observed that:

Table 2: Clustering performance of five methods with respect to different top layer sizes from 20 to 70 on Yale database.

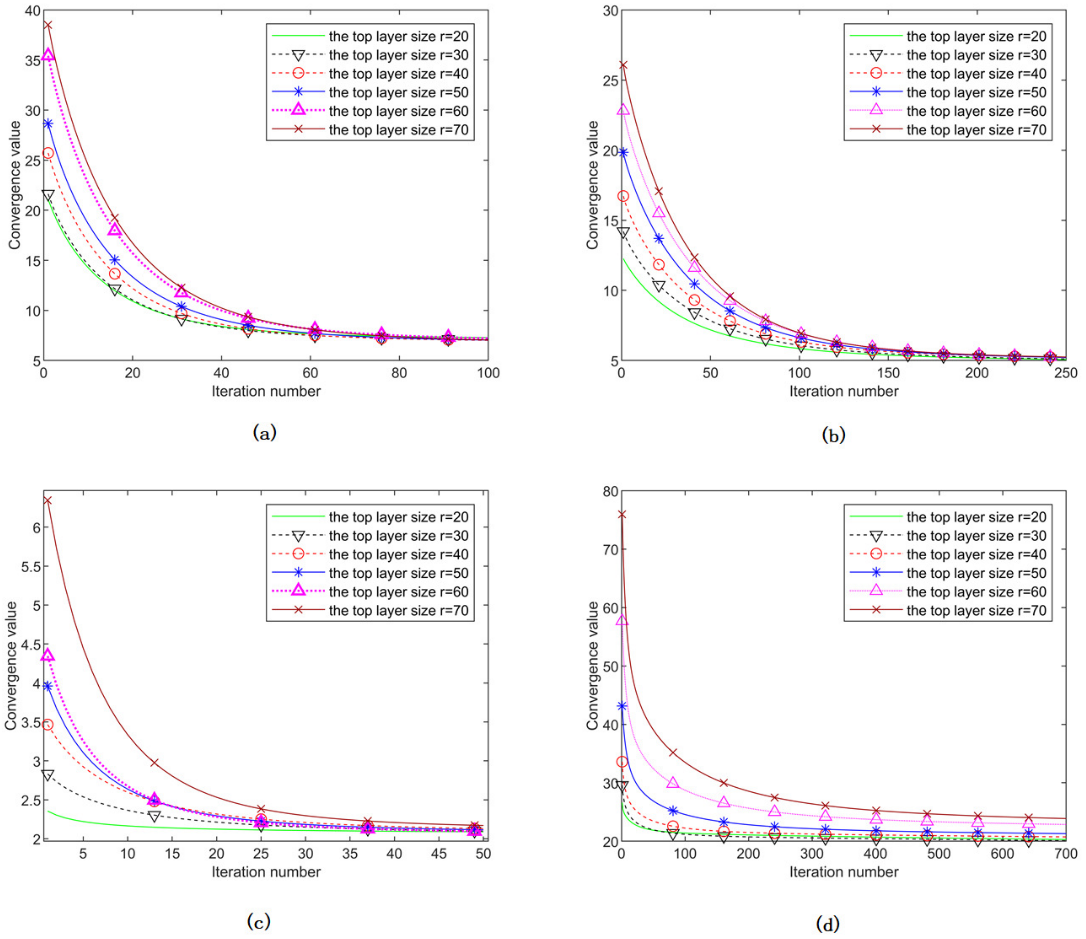
	Methods	20	30	40	50	60	70	Average
ACC(%)	NMF	40.61	36.97	31.52	31.52	35.15	33.94	34.95
	Semi-NMF	43.03	32.73	29.70	22.42	19.39	17.58	27.48
	Deep Semi-NMF	**45.45**	30.30	29.70	30.30	23.03	21.21	30.00
	Deep nsNMF	45.21	40.73	35.76	32.73	34.55	27.88	36.14
	DGsnMF(Ours)	43.03	**44.85**	**47.27**	**46.06**	**47.27**	**43.03**	**45.25**
NMI(%)	NMF	46.80	45.17	41.11	39.34	40.58	43.20	42.70
	Semi-NMF	50.86	39.28	35.56	25.12	20.98	19.92	31.95
	Deep Semi-NMF	**51.46**	39.62	36.97	36.46	28.68	28.35	36.92
	Deep nsNMF	50.36	45.45	39.59	38.21	41.46	33.90	41.50
	DGsnMF(Ours)	50.21	**50.30**	**51.88**	**52.04**	**54.13**	**48.03**	**51.10**

Table 3: Clustering performance of five methods with respect to different top layer sizes from 20 to 70 on ORL database.

	Methods	20	30	40	50	60	70	Average
ACC(%)	NMF	41.50	45.75	44.25	37.25	39.25	40.50	41.42
	Semi-NMF	56.00	56.75	49.50	44.50	39.75	34.74	46.87
	Deep Semi-NMF	55.00	57.00	58.00	51.75	46.00	44.25	52.00
	Deep nsNMF	54.75	56.50	57.25	53.25	52.50	51.00	54.21
	DGsnMF(Ours)	**59.00**	**65.50**	**63.00**	**59.25**	**63.50**	**62.75**	**62.17**
NMI(%)	NMF	64.72	67.29	66.41	60.76	62.10	63.07	64.06
	Semi-NMF	74.79	74.36	72.52	66.18	61.74	56.80	67.73
	Deep Semi-NMF	75.95	77.31	75.18	71.82	70.41	66.50	72.86
	Deep nsNMF	75.50	76.53	77.74	75.94	74.65	72.94	75.55
	DGsnMF(Ours)	**78.44**	**78.41**	**78.83**	**77.78**	**80.25**	**79.61**	**78.89**

Table 4: Clustering performance of five methods with respect to different top layer sizes from 20 to 70 on JAFFE database.

	Methods	20	30	40	50	60	70	Average
ACC(%)	NMF	84.98	82.63	85.45	76.53	79.81	77.00	81.07
	Semi-NMF	77.00	52.58	39.91	28.64	24.88	27.23	41.71
	Deep Semi-NMF	92.02	77.46	61.50	49.77	47.89	35.68	60.72
	Deep nsNMF	93.90	68.08	78.87	54.93	40.38	52.58	64.79
	DGsnMF(Ours)	**98.12**	**95.77**	**95.31**	**96.71**	**95.31**	**94.84**	**96.01**
NMI(%)	NMF	82.45	80.45	81.73	78.53	77.14	80.94	80.21
	Semi-NMF	76.50	51.21	41.93	26.85	17.33	17.61	38.57
	Deep Semi-NMF	89.65	76.42	68.75	55.44	48.71	33.83	62.13
	Deep nsNMF	91.68	78.52	82.78	68.49	60.67	65.69	74.64
	DGsnMF(Ours)	**96.96**	**93.54**	**92.96**	**95.14**	**92.71**	**92.41**	**93.95**



(a)

(b)

(c)

(d)

Figure 3: The performance of convergence value with iteration numbers of top layer sizes (from 20 to 70) in four datasets: (a) Yale. (b) ORL. (c) JAFFE. (d) PIE.

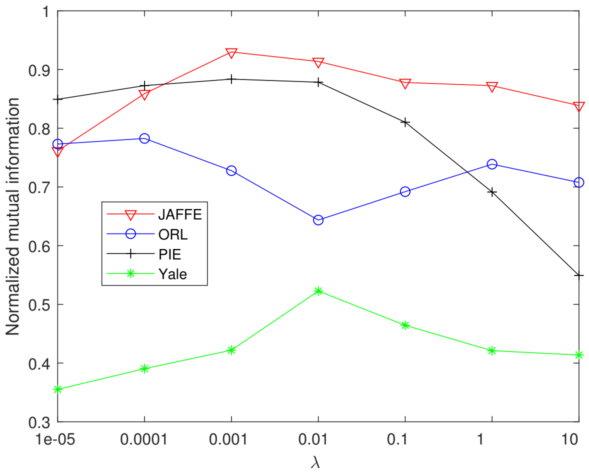


Figure 4: Normalized mutual information of DGsnMF model with different hyper-parameters λ on four datasets.

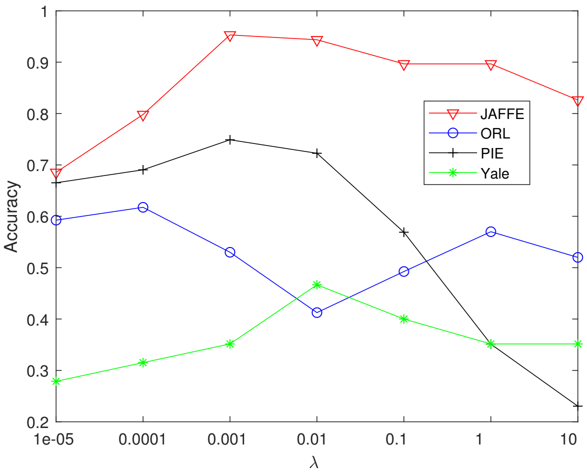


Figure 5: Clustering accuracy of DGsnMF model with different hyper-parameters λ on four datasets.

1. Compared with NMF and Semi-NMF which are single layer models, the deep architectures algorithm (Deep Semi-NMF, Deep nsNMF, DGsnMF) can achieve the better clustering performance, which means multi-layer structure methods have clustering advantages over single-layer methods. It is clear that our DGsnMF model always

Table 5: Clustering performance of five methods with respect to different top layer sizes from 20 to 70 on PIE database.

	Methods	20	30	40	50	60	70	Average
ACC(%)	NMF	41.63	41.91	53.54	50.14	56.09	57.67	50.16
	Semi-NMF	62.22	64.86	70.48	73.95	76.77	76.84	70.85
	Deep Semi-NMF	65.09	68.66	76.05	78.86	80.91	79.18	74.79
	Deep nsNMF	65.02	69.86	71.89	74.66	77.07	74.65	72.19
	DGsnMF(Ours)	**67.26**	**76.33**	**76.30**	**84.35**	**84.91**	**84.87**	**79.00**
NMI(%)	NMF	71.43	74.21	79.34	79.98	82.89	82.61	78.41
	Semi-NMF	80.05	83.54	84.57	88.65	90.78	90.32	86.32
	Deep Semi-NMF	83.54	85.57	88.97	91.63	93.16	92.10	89.16
	Deep nsNMF	82.14	85.95	87.96	89.36	91.78	90.23	87.90
	DGsnMF(Ours)	**85.11**	**90.40**	**92.48**	**95.86**	**96.67**	**96.88**	**92.91**

gets the best result in different datasets experiments which means that DGsnMF has good robustness to solve different data distribution problems.

2. By comparing the performance with respect to different top layer sizes from 20 to 70, we can observe that the proposed DGsnMF has better clustering results than others on the four datasets in most of cases, although Deep Semi-NMF performs better when the top layer size is 20 in Yale dataset in Table 2. From Table 2 and Table 4, we can find that the clustering performance of the Semi-NMF and Deep Semi-NMF become worse as the layer sizes increase. Our DGsnMF is robust to the change of top layer size, which can facilitate the tuning of the models in practical applications.

5. Conclusion

In this paper, we proposed a novel unsupervised non-negative matrix factorization method named DGsnMF to learn to learn the low-dimensional compact representations for clustering. DGsnMF adapts the deep matrix factorization to a deep learning approach to automatically learn a hierarchy of attributes of data, and introduces the graph regularization term to leverage the geometrical information. More importantly, we employ the forward-backward splitting framework to solve the nonconvex optimization problem in DGsnMF. Not only is our proposed algorithm computationally efficient in practice, but also it has strong convergence guaranteed by theoretical and experimental analysis. From all the clustering results, we can see that DGsnMF has better clustering results than the four existing methods on several public benchmark image datasets. In the future, we will focus on improving the robustness of our model to noise and outliers, and extend it to the application of multi-view clustering. The recent relative application using deep matrix factorization models have been published in [25] and [26], but their solutions approximate nonconvex constraints using convex constraints.

Appendix A.

At first, we define the critical points for nonconvex and nonsmooth functions, semi-algebraic functions and KL functions used for the convergence analysis. Definition 1: For a proper and lower semi-continuous function h,

- The *Fréchet subdifferential* of h at x can be written as $\hat{\partial} h(x)$ is a set $\mathbf{u}$ and satisfies the following equation if $x \in$ dom h

$$\left\{\mathbf{u} \in \mathbb{R}^d : \liminf_{y \to x \ y \neq x} \frac{h(y) - h(x) - \langle u, y - x \rangle}{\|y - x\|} \geq 0 \right\} \tag{A.1}$$

  and $\hat{\partial} h(x) = \emptyset$ when $x \notin$ dom h

- The limiting subdifferental of h at x, written as $\partial h(x)$, is a set $\mathbf{u}$ defined as follows:

$$\left\{\mathbf{u} \in \mathbb{R}^d : \exists x^k \to x, h(x^k) \to h(x), \mathbf{u}^k \in \hat{\partial} h(x) \to \mathbf{u}, k \to \infty \right\} \tag{A.2}$$

- Point x satisfies the following equation and it is called limiting-critical point of function h:

$$0 \in \partial h(x) \tag{A.3}$$

Definition 2: (Kurdyka-Łojasiewicz Property): A function h has the KL property at $x^* \in$ dom ∂h if there exist:

1. $\eta \in (0, \infty)$, $s \in (0, \eta)$,a neighborhood V of x^*
2. a continuous concave function $\varphi : [0, \eta) \to \mathbb{R}_+$,such that the following holds:

   a). φ is C^1 on $(0, \eta)$, $\varphi(0) = 0$ and $\varphi'(s) > 0$

   b). $\forall x \in V$ satisfies $h(x^*) < h < h(x^* + \eta)$, the KL inequality holds:

$$\varphi' \left(h(x) - h\left(x^*\right)\right) \operatorname{dist}(0, \partial h(x)) \geq 1 \tag{A.4}$$

3. Function h satisfies the KL property at each dom ∂h

Definition 3:(Semi-algerbraic sets and functions[16])

- A subset S of $\mathbb{R}^n$ is called semialgerbraic set if there exists a finite number of real polynomial functions g_{ij}, g'_{ij}: $\mathbb{R}^n \to \mathbb{R}$ such that

$$S = \bigcup_{j=1} \bigcap_{i=1} \{x \in \mathbb{R}^n : g_{ij}(x) = 0, g'_{ij}(x) < 0\} \tag{A.5}$$

- A function $h : \mathbb{R}^n \to (-\infty, +\infty]$ is semialgebraic if its graph $\{(x, y) \in \mathbb{R}^n \times \mathbb{R}, y = h(x)\}$ is a semialgebraic set.

Then following the conditions as described in Theorem 3.2, we check the sequence $\{\mathbf{J}^k\} = \{\mathbf{W}_1^k, \mathbf{W}_2^k, ..., \mathbf{W}_l^k, \mathbf{H}_l^k\}$ generated by the Algorithm 1 satisfies the conditions V1-V4.

Condition V1: Proof: Forward-backward splitting framework is to solve the DGsnMF problem as described in (9). As for weighted matrix $\mathbf{W}_i$ ($i \in 1, 2, .., l$), it is updated through the iteration as described in (17) and we have

$$\begin{aligned} &\frac{\mu_i^k}{2}\left\|\mathbf{W}_i^{k+1} - \left(\mathbf{W}_i^k - \frac{1}{\mu_i^k}\nabla_{\mathbf{W}_i} Q(\mathbf{W}_1^{k+1}, ..., \mathbf{W}_i^k, ..., \mathbf{W}_l^k, \mathbf{H}_l^k)\right)\right\|_F^2 \\ &\qquad + F(\mathbf{W}_i^{k+1}) \\ &\le \frac{\mu_i^k}{2}\|\frac{1}{\mu_i^k}\nabla_{\mathbf{W}_i} Q(\mathbf{W}_1^{k+1}, ..., \mathbf{W}_i^k, ..., \mathbf{W}_l^k, \mathbf{H}_l^k)\|_F^2 + F(\mathbf{W}_i^k) \end{aligned} \tag{A.6}$$

then,

$$\begin{aligned} F(\mathbf{W}_i^k) \ge F(\mathbf{W}_i^{k+1}) + \frac{\mu_i^k}{2}\|\mathbf{W}_i^{k+1} - \mathbf{W}_i^k\|_F^2 \\ + trace\langle \mathbf{W}_i^{k+1} - \mathbf{W}_i^k, \nabla_{\mathbf{W}_i} Q(\mathbf{W}_1^{k+1}, ..., \mathbf{W}_i^k, ..., \mathbf{W}_l^k, \mathbf{H}_l^k)\rangle \end{aligned} \tag{A.7}$$

It is noteworthy that $\nabla_{\mathbf{W}_i} Q$ is Lipschitz continuous and Q is the C^1 function. Let $L_{\mathbf{W}_i}$ denote the Lipschitz constant of gradient $\nabla_{\mathbf{W}_i} Q$ and we have

$$\begin{aligned} Q(\mathbf{W}_1^{k+1}, ..., \mathbf{W}_i^{k+1}, ..., \mathbf{W}_l^k, \mathbf{H}_l^k) \le \\ Q(\mathbf{W}_1^{k+1}, ..., \mathbf{W}_i^k, ..., \mathbf{W}_l^k, \mathbf{H}_l^k) \\ + trace\langle \mathbf{W}_i^{k+1} - \mathbf{W}_i^k, \nabla_{\mathbf{W}_i} Q(\mathbf{W}_1^{k+1}, ..., \mathbf{W}_i^k, ..., \mathbf{W}_l^k, \mathbf{H}_l^k)\rangle \\ + \frac{L_{\mathbf{W}_i}}{2}\|\mathbf{W}_i^{k+1} - \mathbf{W}_i^k\|_F^2 \end{aligned} \tag{A.8}$$

Combining (A.7) and (A.8), we obtain the following inequality:

$$\begin{aligned} F(\mathbf{W}_i^{k+1}) + Q(\mathbf{W}_1^{k+1}, ..., \mathbf{W}_i^{k+1}, ..., \mathbf{W}_l^k, \mathbf{H}_l^k) \le F(\mathbf{W}_i^k) \\ + Q(\mathbf{W}_1^{k+1}, ..., \mathbf{W}_i^k, ..., \mathbf{W}_l^k, \mathbf{H}_l^k) - \frac{\mu_i^k - L_{\mathbf{W}_i}}{2}\|\mathbf{W}_i^{k+1} - \mathbf{W}_i^k\|_F^2 \end{aligned} \tag{A.9}$$

Similarly, as for feature matrix $\mathbf{H}_l$, we have,

$$\begin{aligned} Q(\mathbf{W}_1^{k+1}, ..., \mathbf{W}_i^{k+1}, ..., \mathbf{W}_l^{k+1}, \mathbf{H}_l^{k+1}) + G(\mathbf{H}_l^{k+1}) \le G(\mathbf{H}_i^k) \\ + Q(\mathbf{W}_1^{k+1}, \mathbf{W}_2^{k+1}, ..., \mathbf{W}_l^{k+1}, \mathbf{H}_l^k) - \frac{\nu^k - L_{\mathbf{H}_l}}{2}\|\mathbf{H}_l^{k+1} - \mathbf{H}_l^k\|_F^2 \end{aligned} \tag{A.10}$$

The step sizes sequences μ_i^k and ν^k are chosen as $0 < \frac{1}{\mu_i^k} \le \frac{1}{L_Q}$ and $0 < \frac{1}{\nu^k} \le \frac{1}{L_Q}$. Summing up (A.9) and (A.10), we can thereby obtain the following equation:

$$\begin{aligned} \phi(\mathbf{W}_1^k, \mathbf{W}_2^k, ..., \mathbf{W}_l^k, \mathbf{H}_l^k) - \phi(\mathbf{W}_1^{k+1}, ..., \mathbf{W}_i^{k+1}, ..., \mathbf{W}_l^{k+1}, \mathbf{H}_l^{k+1}) \\ \ge \sum_{i=1}^{l} \frac{\mu_i^k - L_{\mathbf{W}_i}}{2}\|\mathbf{W}_i^{k+1} - \mathbf{W}_i^k\|_F^2 + \frac{\nu^k - L_{\mathbf{H}_l}}{2}\|\mathbf{H}_l^{k+1} - \mathbf{H}_l^k\|_F^2 \end{aligned} \tag{A.11}$$

Let $a_1 = (\mu_i^k - L_{\mathbf{W}_i})/2$ and $a_2 = (\nu^k - L_{\mathbf{H}_l})/2$. There exists $a := (a_1, a_2) > 0$ since $\mu_i^k > L_Q$ and $\nu^k > L_Q$. Thus, $\{\mathbf{J}^k\} = \{\mathbf{W}_1^k, \mathbf{W}_2^k, ..., \mathbf{W}_l^k, \mathbf{H}_l^k\}$ satisfy the condition V1. ■

Condition V2: Proof: The iteration of weighted matrix $\mathbf{W}_i$ is presented as follows:

$$\mathbf{W}_i^{k+1} = \arg\min_{\mathbf{W}_i} F(\mathbf{W}_i) + \hat{Q}(\mathbf{W}_i) + \frac{\mu_i^k}{2}\|\mathbf{W}_i - \mathbf{W}_i^{k+1}\|_F^2 \tag{A.12}$$

where $\hat{Q}(\mathbf{W}_i) = Q(\mathbf{W}_1^{k+1}, ..., \mathbf{W}_i^k, ..., \mathbf{W}_l^k, \mathbf{H}_l^k) + trace\langle \mathbf{W}_i - \mathbf{W}_i^{k+1}, \nabla_{\mathbf{H}_l} Q(\mathbf{W}_1^{k+1}, ..., \mathbf{W}_i^k, ..., \mathbf{W}_l^k, \mathbf{H}_l^k)\rangle$ Therefore, we have

$$\begin{aligned} 0 \in \partial F(\mathbf{W}_i^{k+1}) + \nabla_{\mathbf{W}_i} Q(\mathbf{W}_1^{k+1}, ..., \mathbf{W}_i^k, ..., \mathbf{W}_l^k, \mathbf{H}_l^k) \\ + \mu_i^k(\mathbf{W}_i^{k+1} - \mathbf{W}_i^k) \end{aligned} \tag{A.13}$$

Then

$$\begin{aligned} \nabla_{\mathbf{W}_i} Q(\mathbf{J}^{k+1}) - \nabla_{\mathbf{W}_i} Q(\mathbf{W}_1^{k+1}, ..., \mathbf{W}_i^k, ..., \mathbf{W}_l^k, \mathbf{H}_l^k) \\ - \mu_i^k(\mathbf{W}_i^{k+1} - \mathbf{W}_i^k) \in \partial_{\mathbf{W}_i}\phi(\mathbf{J}^{k+1}) \end{aligned} \tag{A.14}$$

For the feature matrix $\mathbf{H}_l$, the iteration has been presented in (16), we can obtain:

$$\begin{aligned} \nabla_{\mathbf{H}_l} Q(\mathbf{J}^{k+1}) - \nabla_{\mathbf{H}_l} Q(\mathbf{W}_1^{k+1}, ..., \mathbf{W}_i^{k+1}, ..., \mathbf{W}_l^{k+1}, \mathbf{H}_l^k) \\ - \nu^k(\mathbf{H}_l^{k+1} - \mathbf{H}_l^k) \in \partial_{\mathbf{H}_l}\phi(\mathbf{J}^{k+1}) \end{aligned} \tag{A.15}$$

Define: $\mathbf{N}^k := (\mathbf{N}_{\mathbf{W}_i}^k, \mathbf{N}_{\mathbf{H}_l}^k)$

$$\begin{aligned} \mathbf{N}_{\mathbf{W}_i}^{k+1} := \nabla_{\mathbf{W}_i} Q(\mathbf{J}^{k+1}) - \nabla_{\mathbf{W}_i} Q(\mathbf{W}_1^{k+1}, ..., \mathbf{W}_i^k, ..., \mathbf{W}_l^k, \mathbf{H}_l^k) \\ - \mu_i^k(\mathbf{W}_i^{k+1} - \mathbf{W}_i^k) \in \partial_{\mathbf{W}_i}\phi(\mathbf{J}^{k+1}) \end{aligned} \tag{A.16}$$

$$\begin{aligned} \mathbf{N}_{\mathbf{H}_l}^{k+1} := \nabla_{\mathbf{H}_l} Q(\mathbf{J}^{k+1}) - \nabla_{\mathbf{H}_l} Q(\mathbf{W}_1^{k+1}, ..., \mathbf{W}_i^{k+1}, ..., \mathbf{W}_l^{k+1}, \mathbf{H}_l^k) \\ - \nu^k(\mathbf{H}_l^{k+1} - \mathbf{H}_l^k) \in \partial_{\mathbf{H}_l}\phi(\mathbf{J}^{k+1}) \end{aligned} \tag{A.17}$$

If $\{\mathbf{J}^k\}$ is bounded, since ∇Q is Lipschitz continuous on any bounded set. Then $\mathbf{N}^k := (\mathbf{N}_{\mathbf{W}_i}^k, \mathbf{N}_{\mathbf{H}_l}^k) \in \partial\phi(\mathbf{J}^k)$, exist positive constants $p > 0$ and satisfy the following inequality:

$$\|\mathbf{N}^{k+1}\|_F^2 \le p\|\mathbf{J}^{k+1} - \mathbf{J}^k\|_F^2 \tag{A.18}$$

and condition V2 is proved. ■

Condition V3: Proof: From V1, $\phi(\mathbf{J})$ is decreasing and $\inf\{\phi\} > -\infty$. Given a positive integer j, we can obtain the following inequality:

$$\phi(\mathbf{J}^0) - \phi(\mathbf{J}^j) \ge a\sum_{k=0}^{j}\|\mathbf{J}^k - \mathbf{J}^{k+1}\|_F^2 \tag{A.19}$$

Therefore, there exists some $\phi(\tilde{\mathbf{J}})$ such that $\phi(\mathbf{J}^j) \to \phi(\tilde{\mathbf{J}})$ as $j \to +\infty$. When $j \to +\infty$, we have

$$\sum_{k=0}^{j\to+\infty}\|\mathbf{J}^k - \mathbf{J}^{k+1}\|_F^2 < \phi(\mathbf{J}^0) - \phi(\mathbf{J}^j) < +\infty \tag{A.20}$$

which means $\lim\|\mathbf{J}^k - \mathbf{J}^{k+1}\|_F^2 = 0$.
For weight matrices $\mathbf{W}_i$, from (A.12), we assume $k = k_j - 1 (j \to +\infty)$ and $\mathbf{W}_i = \tilde{\mathbf{W}}_i$ and obtain

$$\begin{aligned} &F(\mathbf{W}_i^{k_j}) + \hat{Q}(\mathbf{W}_i^{k_j}) + \frac{\mu_i^{k_j-1}}{2}\|\mathbf{W}_i^{k_j} - \mathbf{W}_i^{k_j-1}\|_F^2 \\ &\leq F(\tilde{\mathbf{W}}_i) + \hat{Q}(\tilde{\mathbf{W}}_i) + \frac{\mu_i^{k_j-1}}{2}\|\tilde{\mathbf{W}}_i - \mathbf{W}_i^{k_j-1}\|_F^2 \end{aligned} \tag{A.21}$$

From condition V1 and Lipschitz continuity of ∇Q, we can obtain

$$\lim\ \sup_{j\to+\infty} F(\mathbf{W}_i^{k_j}) \leq F(\tilde{\mathbf{W}}_i) \tag{A.22}$$

Similarly, for feature matrix $\mathbf{H}_l$, we get

$$\lim\ \sup_{j\to+\infty} G(\mathbf{H}_l^{k_j}) \leq G(\tilde{\mathbf{H}}_l) \tag{A.23}$$

It should be noted that function F and G are lower semi-continuous, we have,

$$\lim_{j\to+\infty} F(\mathbf{W}_i^{k_j}) = F(\tilde{\mathbf{W}}_i), \lim_{j\to+\infty} G(\mathbf{H}_l^{k_j}) = G(\tilde{\mathbf{H}}_l) \tag{A.24}$$

Moreover, Q is continuous, we can thereby obtain the following equation:

$$\lim_{j\to+\infty} \phi(\mathbf{J}^{k_j}) = \phi(\tilde{\mathbf{J}}), \tag{A.25}$$

and complete the proof of condition V3. ■
Theorem [17]: Proper closed semi-algebraic function h satisfies the KL property at any point in dom h.
Condition V4: Proof: Q is a real polynomial function, so it is naturally a semialgebraic function. The graph of indicator function δ_S can be rewritten as $\{(u, 0) : u \in S\} \cup \{(v, 0) : v \in \bar{S}\}$, set S_F and S_G are both nonempty closed semialgebraic sets according to their definitions, therefore, functions F and G are semialgebraic . Our objective function $\phi(\mathbf{W}_1, ..., \mathbf{W}_l, \mathbf{H}_l) = Q(\mathbf{W}_1, ..., \mathbf{W}_l, \mathbf{H}_l) + I(\mathbf{W}_1, ..., \mathbf{W}_l, \mathbf{H}_l)$ defined in (9) satisfies the KL property. ■

Acknowledgments

This work was supported in part by the National Natural Science Foundation of China under Grant 61803096.

References

[1] D. D. Lee, and H. S. Seung, “Learning the parts of objects by non-negative matrix factorization,” *Nature*, vol. 401, no. 6755, pp. 788-791, Oct. 1999.

[2] D. Cai, X. He, J. Han, and T. S. Huang, “Graph regularized nonnegative matrix factorization for data representation,” *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 33, no. 8, pp. 1548-1560, Dec. 2010.

[3] C. H. Ding, T. Li, and M. I. Jordan, “Convex and semi-nonnegative matrix factorizations,” *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 32, no. 1, pp. 45-55, Nov. 2008.

[4] Y. LeCun, Y. Bengio, and G. Hinton, “Deep learning,” *Nature*, vol. 521, no. 7553, pp. 436-444, May 2015.

[5] H. A. Song, B.-K. Kim, T. L. Xuan, and S.-Y. Lee, “Hierarchical feature extraction by multi-layer non-negative matrix factorization network for classification task,” *Neurocomputing*, vol. 165, pp. 63-74, Oct. 2015.

[6] G. Trigeorgis, K. Bousmalis, S. Zafeiriou, and B. W. Schuller, “A deep matrix factorization method for learning attribute representations,” *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 39, no. 3, pp. 417-429, Mar. 2017.

[7] J. Yu, G. Zhou, A. Cichocki, and S. Xie, “Learning the hierarchical parts of objects by deep non-smooth nonnegative matrix factorization,” *IEEE Access*, vol. 6, pp. 58096-58105, Oct. 2018.

[8] H.-C. Li, G. Yang, W. Yang, Q. Du, and W. J. Emery, “Deep nonsmooth nonnegative matrix factorization network factorization network with semi-supervised learning for SAR image change detection,” *ISPRS Journal of Photogrammetry and Remote Sensing*, vol. 160, pp. 167-179, Feb. 2020.

[9] X.-R. Feng, H.-C. Li, J. Li, Q. Du, A. Plaza and W. J. Emery, “Hyperspectral unmixing using sparsity-constrained deep nonnegative matrix factorization with total variation,” *IEEE Trans. Geoscience and Remote Sensing*, vol. 56, no. 10, pp. 6245-6257, Jun. 2018.

[10] Z. Li, S. Ding, Y. Li, Z. Yang, S. Xie and W. Chen, “Manifold optimization-based analysis dictionary learning with an $\ell 1/2$-norm regularizer,” *Neural Networks*, vol. 98, pp. 212-222, Feb. 2018.

[11] H. Zhao, Z. Ding, and Y. Fu, “Multi-view clustering via deep matrix factorization,” *In Thirty-First AAAI Conf. Artificial Intelligence*, San Francisco, USA, Feb. 2017, pp. 1288-1293.

[12] H. Fang, A. Li, H. Xu, and T. Wang, “Sparsity-constrained deep nonnegative matrix factorization for hyperspectral unmixing,” *IEEE Geoscience and Remote Sensing Letters*, vol. 15, no. 7, pp. 1105-1109, Jul. 2018.

[13] F. Ye, C. Chen, and Z. Zheng, “Deep autoencoder-like nonnegative matrix factorization for community detection,” *In Proc. 27th ACM Int. Conf. Information and Knowledge Management*, New York, USA, Oct. 2018, pp. 1393-1402.

[14] T. Zhu,“Sparse dictionary learning by block proximal gradient with global convergence,” *Neurocomputing*, vol. 367, pp. 226-235, Nov. 2019.

[15] G. Peng, “Joint and direct optimization for dictionary learning in convolutional sparse representation,” *IEEE Trans. Neural Networks and Learning Systems*, vol. 31, no. 2, pp. 559-573, Feb. 2020.

[16] H. Attouch, J. Bolte, P. Redont, and A. Soubeyran, “Proximal alternating minimization and projection methods for nonconvex problems: An approach based on the Kurdyka-Łojasiewicz inequality,” *Mathematics of Operations Research*, vol. 35, no. 2, pp. 438-457, Apr. 2010.

[17] J. Bolte, S. Sabach, and M. Teboulle, “Proximal alternating linearized minimization for nonconvex and nonsmooth problems,” *Mathematical Programming*, vol. 146, no. 1-2, pp. 459-494, Aug. 2014.

[18] H. Attouch, J. Bolte, and B. F. Svaiter, “Convergence of descent methods for semi-algebraic and tame problems: proximal algorithms, forward–backward splitting, and regularized Gauss–Seidel methods,” *Mathematical Programming*, vol. 137, no. 1-2, pp. 91-129, Feb. 2013.

[19] Y. Bengio, “Learning deep architectures for AI,” *Foundations and trends® in Machine Learning*, vol. 2, no. 1, pp. 1-127, Nov. 2009.

[20] Z. Li, S. Ding, W. Chen, Z. Yang, and S. Xie, “Proximal alternating minimization for analysis dictionary learning and convergence analysis,” *IEEE Trans. Emerging Topics in Computational Intelligence*, vol. 2, no. 6, pp. 439-449, Dec. 2018.

[21] N. Guan, D. Tao, Z. Luo, and B. Yuan, “Nenmf: An optimal gradient method for nonnegative matrix factorization,” *IEEE Trans. Signal Processing*, vol. 60, no. 6, pp. 2882-2898, Jun. 2012.

[22] M. Lyons, S. Akamatsu, M. Kamachi, and J. Gyoba, “Coding facial expressions with gabor wavelets,” *In Proc. Third IEEE Int. Conf. Automatic Face and Gesture Recognition*, Nara, Japan, Apr. 1998, pp. 200-205.

[23] G. E. Hinton, and R. R. Salakhutdinov, “Reducing the dimensionality of data with neural networks,” *Science*, vol. 313, no. 5786, pp. 504-507, Jul. 2006.

[24] W. Xu, X. Liu, and Y. Gong, “Document clustering based on non-negative matrix factorization,” *In Proc. 26th Annual Int. ACM SIGIR Conf. Research and Development in Informaion Retrieval*, Toronto, Canada, Jul. 2003, pp. 267-273.

[25] S. Huang, Z. Kang, I.W. Tsang, Z. Xu, “Auto-weighted multi-view clustering via kernelized graph learning,” *Pattern Recognition*, vol. 97, no. 107015. Jan. 2020.

[26] J. Li, G. Zhou, Y. Qiu, Y. Wang, Y. Zhang and S. Xie, “Deep graph regularized non-negative matrix factorization for multi-view clustering,” *Neurocomputing*, Dec. 2019. [Online]. Available: https://doi.org/10.1016/j.neucom.2019.12.054