# Generalized Discriminative Deep Non-Negative Matrix Factorization Based on Latent Feature and Basis Learning

**Zijian Yang**[1] , **Zhiwei Li**[1] and **Lu Sun**[1]
[1]ShanghaiTech University
{yangzj, lizhw, sunlu1}@shanghaitech.edu.cn

## Abstract

As a powerful tool for data representation, deep NMF has attracted much attention in recent years. Current deep NMF builds the multi-layer structure by decomposing either basis matrix or feature matrix into multiple factors, and probably complicates the learning process when data is insufficient or exhibits simple structure. To overcome the limitations, a novel method called Generalized Deep Non-negative Matrix Factorization (GDNMF) is proposed, which generalizes several NMF and deep NMF methods in a unified framework. GDNMF simultaneously performs decomposition on both features and bases, which learns a hierarchical data representation based on multi-level basis. To further improve the latent representation and enhance its flexibility, GDNMF mutually reinforces shallow linear model and deep non-linear model. Moreover, semi-supervised GDNMF is proposed by treating partial label information as soft constraints in the multi-layer structure. An efficient two-phase optimization algorithm is developed, and experiments on five real-world datesets verify its superior performance compared with state-of-the-art methods.

## 1 Introduction

Data processing and analysis are of great importance in artificial intelligence area, and have a wide range of applications, such as face recognition [Karczmarek *et al.*, 2019], computer vision [Buckler *et al.*, 2018] and signal processing [Baraniuk, 2011]. However, a large amount of data collected from realword applications is often accompanied with irrelevant and redundant information, and the data usually exhibits high-dimensionality. Such data probably results in a high computational cost in applications, and harms the generalization ability of a learning algorithm. To recover the latent representation and reduce the redundancy, various dimension reduction methods have been proposed, such as Principal Component Analysis (PCA) [Moore, 1981], Locally Preserving Projection (LPP) [He and Niyogi, 2003], Linear Discriminant Analysis (LDA) [Li and Yuan, 2004] and Non-negative Matrix Factorization (NMF) [Lee and Seung, 1999].

As a popular and promising dimensionality reduction method, NMF decomposes a non-negative data matrix $\mathbf{X}$ into a product of two non-negative matrices: the latent basis matrix $\mathbf{W}$ and the latent feature matrix $\mathbf{H}$, and treats the feature matrix as the latent attribute representation of the original data. Due to the non-negativity constraints, NMF enables to learn part-based representation by only allowing additive combination, that leads to better interpretability in practice. As a variant of NMF, semi-NMF [Ding *et al.*, 2008] relaxes non-negativity constraints on the basis matrix and the data matrix, which extends the applicability of NMF when the data is not strictly positive. It can be interpreted from the clustering perspective: the basis matrix contains cluster centroids, while the feature matrix indicates the cluster assignment. In addition to the above unsupervised NMF methods, semi-supervised NMF seeks to utilize the supervised prior label information in data. According to different ways of using the supervised information, these methods can be roughly categorized into hard constriant methods and soft constriant methods [Chen *et al.*, 2022]. The hard constraint methods [Liu *et al.*, 2011; Meng *et al.*, 2019] learn exactly same representation for samples with same labels. In contrast, the soft constraint methods [Babaee *et al.*, 2016; Lan *et al.*, 2014] utilize partial label information by soft regularization, that allows for similar but distinct representations for samples associated with same labels.

The above methods essentially employ a single-layer structure, that only takes the shallow information of data into account. However, it is possible that data contains complex hierarchical structure information, which conventional single-layered methods cannot extract. Therefore, deep NMF in a multi-layer structure has been proposed [Wisdom *et al.*, 2017; Liu *et al.*, 2017]. These methods build the multi-layer network by further decomposing either the feature matrix or the basis matrix into multiple components [Trigeorgis *et al.*, 2016; Zhao *et al.*, 2021]. They extract latent hierarchical features from the complex data, and obtain a multi-layer feature or basis representation. In addition, semi-supervised deep NMF [Trigeorgis *et al.*, 2016; Meng *et al.*, 2019] is proposed to further improve the performance by using a limited number of labels. Much success has been achieved by deep NMF, however, there are still some limitations. 1) Most deep NMF methods focus on feature decomposition, which fails to model high-level and local basis that consists of lowlevel representations, while recently proposed deep basis decomposition ignores the multi-layer feature representation of data. 2) Current NMF-based methods employ either deep or shallow models, but the shallow model cannot capture hierarchical representations, while the deep model does not always perform well when data is insufficient or exhibits simple representation. 3) Most deep NMF [Zhao *et al.*, 2021] models are unsupervised and thus ignore the prior label information. Even though a few semi-supervised deep NMF models [Meng *et al.*, 2019] are proposed, they usually impose hard constraint such that samples from the same class have exactly the same multi-layer representation, which is too restrictive in practice. Some graph regularized methods [Trigeorgis *et al.*, 2016] implement soft constraint, but they need a quadratic complexity w.r.t. the number of samples, making it intractable on large-scale data.

To overcome these limitations, **G**eneralized **D**eep **N**on-Negative **M**atrix **F**actorization (**GDNMF**) is proposed, which generalizes several NMF and Deep NMF methods in a unified framework. GDNMF enables to extract hierarchical feature representations based on multi-level basis, by conducting both feature and basis decomposition (For Limitation 1). To further improve the latent representation, it mutually reinforces the shallow model and the deep model, such that both simple linear information and complex non-linear structure can be saved (For Limitation 2). Moreover, by treating the supervised information as soft constraints in multi-layer data representation, semi-supervised GDNMF is proposed to utilize the labels in a discriminative manner (For Limitation 3). An efficient optimization algorithm is developed in linear complexity w.r.t. the data size, and experimental results on five real-world datasets demonstrate its superior performance compared with state-of-the-art NMF-based methods. The contributions are summarized as follows:

1. A novel Deep NMF method, named GDNMF, is proposed by simultaneously performing deep factorization on features and bases, which enables to learn a hierarchical data representation based on multi-level basis.
2. GDNMF incorporates linear shallow model and non-linear deep model in a single architecture, which is flexible enough to handle various practical applications and unifies several NMF and deep NMF methods.
3. Semi-supervised GDNMF is developed to use the limited label information in a discriminative way.
4. Extensive experiments on various real-world datasets verify the effectiveness of GDNMF.

## 2 Related Works

### 2.1 Non-Negative Matrix Factorization (NMF)

For a non-negative data matrix $\mathbf{X} \in \mathbb{R}_+^{p \times n}$, NMF [Lee and Seung, 1999] decomposes $\mathbf{X}$ into a product of two low-dimensional non-negative matrices $\mathbf{W}$ and $\mathbf{H}$. Semi-NMF [Ding *et al.*, 2008] relaxes the non-negativity constraints on the data matrix $\mathbf{X}$ and the basis matrix $\mathbf{W}$, leading to a wider range of applications. To encode geometrical information, GNMF [Cai *et al.*, 2010] constructs an affinity graph over samples, and obtains improved generalization ability. SeaNMF [Shi *et al.*, 2018] uses a block coordinate descent algorithm, and incorporates the word-context semantic correlations into the model to discover topics for the short texts. IWNS-NMF [Sabzalian and Abolghasemi, 2018] is proposed to find localized and part-based representations. NMF-LCAG [Yi *et al.*, 2019] introduces a locality constrained graph to discover the latent manifold structure of the data, where the weight matrix of graph and low-dimensional features of data can be learned together. KLS-NMF [Peng *et al.*, 2021a] reveals inherent geometric property of the data, by learning local similarity and clustering in a mutually reinforcing way.

For semi-supervised NMF, CNMF [Liu *et al.*, 2011] utilizes the partial label information as hard constriant, making samples from the same class share the same low-dimensional representations, and thus it guarantees that data with the same labels are grouped into the same clusters. Different from CNMF, DNMF [Babaee *et al.*, 2016] utilizes the label information via a discriminative regularization, which does not necessarily map the samples with the same label into an identical latent representation. RDNMF-SLC [Tong *et al.*, 2019] decomposes the data matrix into three matrices: basis matrix, auxiliary matrix and soft label constraint matrix, and learns a discriminative and robust feature representation. PCMF [Chen *et al.*, 2019] seeks to make the data points from the same class more likely to be merged together in the latent space. CSNMF [Peng *et al.*, 2021b] adopts a correntropy based loss function in constrained NMF to suppress the influence of outliers.

### 2.2 Deep NMF

Deep NMF has been proposed to explore hierarchical features from the complex data, which improves performance in terms of data representation and clustering. [Trigeorgis *et al.*, 2016] proposes Deep semi-NMF (DSNMF), which progressively decomposes the original data matrix $\mathbf{X}$ into $m+1$ factors by $\mathbf{X} \approx \mathbf{W_1 W_2} \cdots \mathbf{W_m H_m}$. It automatically learns a hierarchy of representations for clustering. Different from DSNMF, that factorizes the feature matrix layer by layer, [Zhao *et al.*, 2021] proposes the deep nonnegative basis matrix factorization (DNBMF) that performs deep factorization on the underlying basis matrix by $\mathbf{X} = \mathbf{W}_m \mathbf{H}_m \cdots \mathbf{H}_2 \mathbf{H}_1$. To make the basis vectors sparse, RDNBMF [Zhao *et al.*, 2021] imposes a sparse-inducing regularization term on each layer, and achieves better performance in experiments. In GSDNMF [Fang *et al.*, 2018], the $L_1$ regularizers on both endmember and abundance matrices are imposed to promote sparsity, and the graph regularization in each layer is incorporated to remain the geometric structure. PDNMF [Zhao *et al.*, 2022] adds a basis image reconstruction step to the successive steps of factorization, which helps the basis image to maintain robust feature representation. To enhance the discriminativity of learned features, DDSNnet [Wang *et al.*, 2021] adpots similarity measurements between input data and hidden features as a regularization term, which is beneficial to preserve intrinsic information of original data.

Recently, semi-supervised Deep NMF methods have been proposed to take advantage of the limited prior information. Deep Weakly-Supervised Factorization (Deep WSF) [Trigeorgis *et al.*, 2016] constructs a Laplacian matrix based on the prior label knowledge and uses it to regularize the objective function. SGDNMF [Meng *et al.*, 2019] utilizes the dual-hypergraph Laplacian regularization to capture the high-order relations among data points and enforce the partial label information via a label constraint matrix. By introducing both the global loss and the central loss of the soft label constraint matrix, Deep DRNMF-SLC [Tong *et al.*, 2019] acquires a hierarchical and discriminative data representation. Based on hierarchical non-linear feature extraction, JDSNMF [Moon and Lee, 2021] captures shared latent features from the complex multi-omics data.

Existing deep NMF methods focus on decomposing either the feature matrix or the basis matrix, and are typically constrained in either shallow architectures or deep architectures with hard constraints of labels. In contrast, the proposed GDNMF progressively factorizes both features and bases, so that hierarchical feature representation based on multi-level basis can be extracted. Moreover, it incorporates linear shallow model and non-linear deep model in a unified architecture, and is extended to utilize the label prior information in a discriminative manner.

# 3 Preliminary

## 3.1 Notations

The original data is presented as $\mathbf{X} = [\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_n] \in \mathbb{R}^{p\times n}$, where the $i$-th column $\mathbf{x}_i$ denotes a sample with $p$ features. Let $\mathbf{Y} \in \{0,1\}^{c\times n}$ represent the label matrix and $c$ denote the number of classes. In the semi-supervised setting, only the first $q$ samples (columns) of $\mathbf{Y}$ are labeled, and the rest $(n-q)$ columns are zero. Let $\hat{\mathbf{X}}$ denote the reconstructed $\mathbf{X}$ i.e., $\hat{\mathbf{X}} = \mathbf{W}\mathbf{H}$, where $\mathbf{W} \in \mathbb{R}^{p\times k}$ and $\mathbf{H} \in \mathbb{R}^{k\times n}$ denote the basis matrix and the feature matrix, respectively, and $k$ is the latent dimension. $Tr(\cdot)$ denotes the matrix trace, $\|\cdot\|_F$ is the Frobenius norm with $\|\mathbf{X}\|_F = \sqrt{Tr(\mathbf{X}^T\mathbf{X})}$, and $\odot$ denotes the element-wise product.

## 3.2 Deep NMF with Feature Decomposition

DSNMF [Trigeorgis *et al.*, 2016] decomposes the feature matrix $\mathbf{H}$ into $m$ layers according to:

$$\hat{\mathbf{X}} = \mathbf{W}_1\mathbf{W}_2\cdots\mathbf{W}_m\mathbf{H}_m, \tag{1}$$

where $\mathbf{H}_i = \mathbf{W}_{i+1}\mathbf{H}_{i+1},\ i = 1, 2, \cdots, m-1$, $\mathbf{W}_i$ is the basis matrix of the $i$-th layer, and $\mathbf{H}_i$ is the corresponding feature matrix. Compared with shallow NMF, DSNMF enables to capture a better low-dimensional data representation. Non-linear DSNMF introduces a non-linear squashing function between successive layers to approximate the non-linear manifold in the data. However, only the basis matrix $\mathbf{W}_1$ learned in the first layer is directly related to the original data, which only reflects the shallow local information, and thus DSNMF cannot model high-level and local basis.

## 3.3 Deep NMF with Basis Decomposition

Different from deep feature factorization, DNBMF [Zhao *et al.*, 2021] decomposes the basis matrix $\mathbf{W}$ by

$$\hat{\mathbf{X}} = \mathbf{W}_m\mathbf{H}_m\cdots\mathbf{H}_2\mathbf{H}_1, \tag{2}$$

where $\mathbf{W}_i = \mathbf{W}_{i+1}\mathbf{H}_{i+1},\ i = 1, 2, \cdots, m-1$. By performing deep factorization on the basis matrix, it obtains the underlying basis matrix which can reflect the deep local characteristics of the samples. However, since only the feature matrix $\mathbf{H}_1$ in the first layer is directly related to X, it ignores hierarchical data representation.

# 4 Methodology

## 4.1 Deep Feature and Basis Decomposition

Currently, no deep feature decomposition method considers the high-level and local basis, while no deep basis counterpart can capture the hierarchical feature representation. To overcome the limitation, we propose to simultaneously decompose features and bases in deep NMF. Specifically, at the first layer of the deep model, we decompose the data matrix $\mathbf{X} \in \mathbb{R}^{p\times n}$ into three matrices: $\mathbf{F}_1 \in \mathbb{R}^{p\times k_1}$, $\mathbf{S}_1 \in \mathbb{R}^{k_1\times k_2}$ and $\mathbf{G}_1 \in \mathbb{R}^{k_2\times n}$, and reconstruct $\mathbf{X}$ by

$$\begin{aligned} &\hat{\mathbf{X}} = \mathbf{F}_1\mathbf{S}_1\mathbf{G}_1, \\ &s.t.\quad \mathbf{F}_1 \geq 0, \mathbf{S}_1 \geq 0, \mathbf{G}_1 \geq 0, \end{aligned} \tag{3}$$

where $\mathbf{F}_1$ is the basis matrix, $\mathbf{G}_1$ is the feature matrix and $\mathbf{S}_1$ is an auxiliary matrix that stores the interactions between $\mathbf{F}_1$ and $\mathbf{G}_1$. Then the following deep factorization can be conducted for both basis matrix $\mathbf{F}_1$ and feature matrix $\mathbf{G}_1$. For basis decomposition, we iteratively decompose the basis matrix $\mathbf{F}_1$ layer by layer, according to $\mathbf{F}_i \approx \mathbf{F}_{i+1}\mathbf{S}^L_{i+1}$, where $\mathbf{F}_i$ is the basis matrix at the $i$-th layer, and $\mathbf{S}^L_i$ is the corresponding auxiliary matrix. Thus, the basis decomposition can be expressed as:

$$\begin{aligned} &\hat{\mathbf{F}}_1 = \mathbf{F}_{m_1}\mathbf{S}^L_{m_1}\mathbf{S}^L_{m_1-1}\cdots\mathbf{S}^L_2, \\ &s.t.\quad \mathbf{F}_i \geq 0,\ \mathbf{S}^L_i \geq 0,\ i = 1, 2, \cdots, m_1, \end{aligned} \tag{4}$$

where $m_1$ is the number of layers for basis decomposition. Similarly, for feature decomposition, we iteratively decompose the feature matrix $\mathbf{G}_1$ layer by layer, according to $\mathbf{G}_i \approx \mathbf{S}^R_{i+1}\mathbf{G}_{i+1}$, where $\mathbf{G}_i$ is the feature matrix at the $i$-th layer, and $\mathbf{S}^R_i$ is the corresponding auxiliary matrix. Thus, the feature decomposition can be expressed as:

$$\begin{aligned} &\hat{\mathbf{G}}_1 = \mathbf{S}^R_2\mathbf{S}^R_3\cdots\mathbf{S}^R_{m_2}\mathbf{G}_{m_2}. \\ &s.t.\quad \mathbf{G}_i \geq 0,\ \mathbf{S}^R_i \geq 0,\ i = 1, 2, \cdots, m_2, \end{aligned} \tag{5}$$

where $m_2$ is the number of layers for feature decomposition.

Since real-word data may exhibit deep non-linear hierarchical structures, the linear decomposition may fail to capture possible non-linearity among different levels of the latent features and bases. To address this problem and enhance the model's expressibility, a non-linear function $g(\cdot)$ [1] is introduced for the transformation between successive layers, i.e. $\hat{\mathbf{F}}_i = g(\mathbf{F}_{i+1}\mathbf{S}^L_{i+1})$, and $\hat{\mathbf{G}}_i = g(\mathbf{S}^R_{i+1}\mathbf{G}_{i+1})$. Then the optimization problem is formulated as:

$$\begin{aligned} &\min_{\mathbf{F},\mathbf{S},\mathbf{G}\geq 0} \frac{1}{2}\|\mathbf{X} - \hat{\mathbf{F}}_1\mathbf{S}_1\hat{\mathbf{G}}_1\|^2_F, \\ &s.t.\quad \hat{\mathbf{F}}_1 = g(g(g(\mathbf{F}_{m_1}\mathbf{S}^L_{m_1})\mathbf{S}^L_{m_1-1})\cdots\mathbf{S}^L_2), \\ &\qquad\ \ \hat{\mathbf{G}}_1 = g(\mathbf{S}^R_2 g(\cdots\mathbf{S}^R_{m_2-1}g(\mathbf{S}^R_{m_2}\mathbf{G}_{m_2}))). \end{aligned} \tag{6}$$

[1] Some commonly used non-linear functions include $x^2$, sigmoid($x$) and tanh($x$). In experiments, we set $g(x) = x^2$.

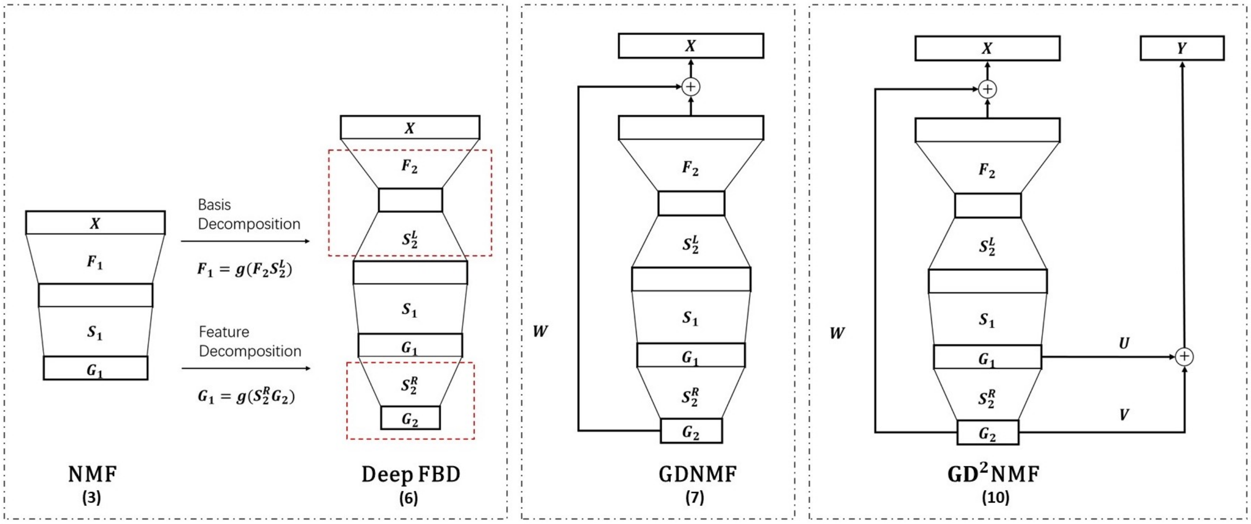

Figure 1: Example of the frameworks of the proposed method. For simplicity, we employ a two-layer structure for both feature and basis factorization. NMF in (3) reconstructs $\mathbf{X}$ by $\hat{\mathbf{X}} = \mathbf{F}_1\mathbf{S}_1\mathbf{G}_1$, and Deep FBD in (6) simultaneously conducts decomposition on feature $\mathbf{G}_1$ and basis $\mathbf{F}_1$ by $\hat{\mathbf{G}}_1 = g(\mathbf{S}_2^R\mathbf{G}_2)$ and $\hat{\mathbf{F}}_1 = g(\mathbf{F}_2\mathbf{S}_2^L)$, respectively. GDNMF in (7) uses the summation of deep nonlinear $\hat{\mathbf{F}}_1\mathbf{S}_1\hat{\mathbf{G}}_1$ and shallow linear $\mathbf{W}\mathbf{G}_2$ to reconstruct $\mathbf{X}$. GD$^2$NMF in (10) utilizes the prior label information $\mathbf{Y}$ based on the reconstruction $\hat{\mathbf{Y}} = \mathbf{U}\hat{\mathbf{G}}_1 + \mathbf{V}\mathbf{G}_2$.

The non-linear function $g(\cdot)$ applied between successive layers helps to extract features of classes that are non-linearly separable in the initial input space, which enhances the model's expressibility and improves the data representation. Moreover, Deep Feature and Basis Decomposition (**Deep FBD**) conducts deep decomposition on the basis matrix and the feature matrix simultaneously, which learns a hierarchical data representation based on multi-level basis. Fig. 1 shows an example of Deep FBD.

### 4.2 Incorporating Shallow and Deep Models

Deep NMF shows its performance advantage compared with classical NMF in previous works [Trigeorgis *et al.*, 2016; Zhao *et al.*, 2021], as it can deal with complex data and obtain hierarchical structure of latent representations. However, when data is insufficient or essentially exhibits a simple latent structure, which can be easily modeled by a linear mapping, the deep structure in deep NMF may complicate the learning process, usually at high computational cost. To address this problem, we propose to incorporate shallow linear model and deep non-linear model together in a unified architecture, to capture various types of information during data representation. To this end, instead of directly minimizing the difference between $\mathbf{X}$ and its reconstruction $\hat{\mathbf{F}}_1\mathbf{S}_1\hat{\mathbf{G}}_1$ in (6), an extra linear representation of feature decomposition $\mathbf{G}_{m_2}$ from the last layer is used to approximate the difference $\mathbf{X} - \hat{\mathbf{F}}_1\mathbf{S}_1\hat{\mathbf{G}}_1$. Therefore, we formulate the optimization problem of **G**eneralized **D**eep **N**on-Negative **M**atrix **F**actorization (**GDNMF**) as:

$$\begin{aligned} &\min_{\mathbf{F},\mathbf{S},\mathbf{G},\mathbf{W}\geq 0} \frac{1}{2}\|\mathbf{X} - \hat{\mathbf{F}}_1\mathbf{S}_1\hat{\mathbf{G}}_1 - \mathbf{W}\mathbf{G}_{m_2}\|_F^2, \\ &s.t. \quad \hat{\mathbf{F}}_1 = g(g(g(\mathbf{F}_{m_1}\mathbf{S}_{m_1}^L)\mathbf{S}_{m_1-1}^L)\cdots\mathbf{S}_2^L), \\ &\qquad\;\; \hat{\mathbf{G}}_1 = g(\mathbf{S}_2^R g(\cdots\mathbf{S}_{m_2-1}^R g(\mathbf{S}_{m_2}^R\mathbf{G}_{m_2}))). \end{aligned} \tag{7}$$

In (7), $\hat{\mathbf{F}}_1\mathbf{S}_1\hat{\mathbf{G}}_1$ is obtained by a deep nonlinear model to capture non-linear complex hierarchical information, while $\mathbf{W}\mathbf{G}_m$ is modeled by a shallow linear model to capture linear information, with $\mathbf{W}$ being the linear basis matrix. Obviously, when the non-linear part is omitted, GDNMF becomes NMF [Lee and Seung, 1999]. When the linear part is omitted, GDNMF becomes DSNMF [Trigeorgis *et al.*, 2016] and DNBMF [Zhao *et al.*, 2021] once $m_1 = 1$ and $m_2 = 1$, respectively. In this way, GDNMF unifies and generalizes several existing NMF and Deep NMF methods, and thus it is flexible enough to handle various practical applications.

### 4.3 Semi-Supervised GDNMF

GDNMF actually ignores the prior label information, but utilizing such label information is the key for performance improvement. In this section, we propose **G**eneralized **D**iscriminative **D**eep **N**on-Negative **M**atrix **F**actorization (**GD$^2$NMF**) to use the limited labeled data in the semi-supervised setting.

Similar with the reconstruction of $\mathbf{X}$ in GDNMF, we consider that both deep information and shallow information contribute to the reconstruction of $\mathbf{Y}$. Specifically, $\mathbf{Y}$ is reconstructed by a summation of linear transformations of deep nonlinear data representation $\mathbf{G}_{m_2}$ and shallow linear data representation $\mathbf{G}_1$. To this end, we propose a discriminative regularization that is expressed as:

$$\|(\mathbf{Y} - \mathbf{U}\hat{\mathbf{G}}_{\mathbf{1}} - \mathbf{V}\mathbf{G}_{m_2})\mathbf{Q}\|_F^2, \tag{8}$$

where $\mathbf{U}$ and $\mathbf{V}$ are weight matrices, and $\mathbf{Q}$ is an auxiliary matrix defined as

$$\mathbf{Q} = \begin{bmatrix} \mathbf{I}_{q\times q} & \\ & \mathbf{0} \end{bmatrix}_{n\times n}. \tag{9}$$

In the semi-supervised setting, only the first $q$ samples are associated with labels ($q < n$), and introducing $\mathbf{Q}$ helps (8) to focus on the reconstruction of the labeled ones. By combining (7) and (8), we obtain the optimization problem of GD$^2$NMF:

$$\begin{aligned}\min_{\substack{\mathbf{F},\mathbf{S},\mathbf{G}\geq 0\\ \mathbf{W},\mathbf{U},\mathbf{V}\geq 0}} &= \frac{1}{2}\|\mathbf{X}-\hat{\mathbf{F}}_1\mathbf{S}_1\hat{\mathbf{G}}_1-\mathbf{W}\mathbf{G}_{m_2}\|_F^2\\ &+\frac{\alpha}{2}\|(\mathbf{Y}-\mathbf{U}\hat{\mathbf{G}}_\mathbf{1}-\mathbf{V}\mathbf{G}_{m_2})\mathbf{Q}\|_F^2, \qquad (10)\\ s.t.\quad &\hat{\mathbf{F}}_1=g(g(g(\mathbf{F}_{m_1}\mathbf{S}^L_{m_1})\mathbf{S}^L_{m_1-1})\cdots\mathbf{S}^L_2),\\ &\hat{\mathbf{G}}_1=g(\mathbf{S}^R_2 g(\cdots\mathbf{S}^R_{m_2-1}g(\mathbf{S}^R_{m_2}\mathbf{G}_{m_2}))),\end{aligned}$$

where $\alpha$ is a positive hyperparameter that makes a trade-off between unsupervised information and supervised information. Thanks to a soft constraint regularization, samples from the same class have similar hierarchical latent representations, rather than exactly the same multi-layer representation which is too restrictive in practice. Therefore, GD$^2$NMF not only conducts factorization on features and bases by a joint architecture consisting of both shallow and deep models, but also utilizes label information in a discriminative manner, leading to improved generalization. An example of the frameworks of both GDNMF and GD$^2$NMF is shown in Fig. 1.

# 5 Optimization Algorithm

In this section, we present the optimization algorithm for GD$^2$NMF in (10), and the optimization algorithm of GDNMF can be simply developed by setting $\alpha = 0$. The algorithm is divided into two stages: pre-training and fine-tune. For pre-training, we solve the sub-problems independently for all layers. Afterwards, we initialize the model in (10) by the pre-trained weights and fine-tune it by an alternating algorithm with gradient backpropagation.

## 5.1 Pre-Training

In pre-training, for basis decomposition we solve the following sub-problem layer by layer,

$$\min_{\mathbf{F}_{i+1},\mathbf{S}^L_{i+1}\geq 0}\frac{1}{2}\|\mathbf{F}_i-g(\mathbf{F}_{i+1}\mathbf{S}^L_{i+1})\|_F^2. \qquad (11)$$

It is a non-convex problem with non-negative constraint, so we fix $\mathbf{F}_{i+1}$ to update $\mathbf{S}^L_{i+1}$, and vice versa, leading to

$$\mathbf{F}_{i+1}=\mathbf{F}_{i+1}\odot\frac{g^{-1}(\mathbf{F}_i)(\mathbf{S}^L_{i+1})^T}{\mathbf{F}_{i+1}\mathbf{S}^L_{i+1}(\mathbf{S}^L_{i+1})^T}, \qquad (12)$$

$$\mathbf{S}^L_{i+1}=\mathbf{S}^L_{i+1}\odot\frac{(\mathbf{F}_{i+1})^Tg^{-1}(\mathbf{F}_i)}{(\mathbf{F}_{i+1})^T\mathbf{F}_{i+1}\mathbf{S}^L_{i+1}}. \qquad (13)$$

For feature decomposition, we independently solve a similar sub-problem for the $i$-th layer, i.e.,

$$\min_{\mathbf{G}_{i+1},\mathbf{S}^R_{i+1}\geq 0}\quad\frac{1}{2}\|\mathbf{G}_i-g(\mathbf{S}^R_{i+1}\mathbf{G}_{i+1})\|_F^2. \qquad (14)$$

Similarly, the closed form solutions are:

$$\mathbf{S}^R_{i+1}=\mathbf{S}^R_{i+1}\odot\frac{g^{-1}(\mathbf{G}_i)\mathbf{G}^T_{i+1}}{\mathbf{S}^R_{i+1}\mathbf{G}_{i+1}\mathbf{G}^T_{i+1}}, \qquad (15)$$

$$\mathbf{G}_{i+1}=\mathbf{G}_{i+1}\odot\frac{(\mathbf{S}^R_{i+1})^Tg^{-1}(\mathbf{G}_i)}{(\mathbf{S}^R_{i+1})^T\mathbf{S}^R_{i+1}\mathbf{G}_{i+1}}. \qquad (16)$$

## 5.2 Fine-Tune

In the fine-tune stage, we solve the optimization problem of GD$^2$NMF in (10) based on alternating optimization. Let $\mathcal{O}$ denote the objective value in (10). The algorithm repeats the following steps until convergence.

**Update F**

To optimize (10) over $\mathbf{F}$, we solve the following sub-problem:

$$\begin{aligned}&\min_{\mathbf{F}\geq 0}\frac{1}{2}\|\mathbf{X}-\hat{\mathbf{F}}_1\mathbf{S}_1\hat{\mathbf{G}}_1-\mathbf{W}\mathbf{G}_{m_2}\|_F^2,\\ s.t.\quad &\hat{\mathbf{F}}_1=g(g(g(\mathbf{F}_{m_1}\mathbf{S}^L_{m_1})\mathbf{S}^L_{m_1-1})\cdots\mathbf{S}^L_2),\\ &\hat{\mathbf{G}}_1=g(\mathbf{S}^R_2g(\cdots\mathbf{S}^R_{m_2-1}g(\mathbf{S}^R_{m_2}\mathbf{G}_{m_2}))).\end{aligned} \qquad (17)$$

Based on the chain rule of derivatives, we get the gradient w.r.t. $\mathbf{F}$ for the $i$-th layer,

$$\begin{aligned}\nabla_{\mathbf{F}_i}\mathcal{O}&=\frac{\partial\mathcal{O}_{obj}}{\partial\mathbf{F}_i}=\frac{\partial\mathcal{O}_{obj}}{\partial\mathbf{F}_i\mathbf{S}^L_i}(\mathbf{S}^L_i)^T\\ &=\left[\frac{\partial\mathcal{O}_{obj}}{\partial g(\mathbf{F}_i\mathbf{S}^L_i)}\odot g^{'}(\mathbf{F}_i\mathbf{S}^L_i)\right](\mathbf{S}^L_i)^T \qquad (18)\\ &=\left[\nabla_{\mathbf{F}_{i-1}}\mathcal{O}\odot g^{'}(\mathbf{F}_i\mathbf{S}^L_i)\right](\mathbf{S}^L_i)^T,\end{aligned}$$

and the derivative w.r.t. $\mathbf{F}_1$:

$$\nabla_{\mathbf{F}_1}\mathcal{O}=-(\mathbf{X}-\mathbf{F}_1\mathbf{S}_1\mathbf{G}_1-\mathbf{W}\mathbf{G}_{m_2})\mathbf{G}_1^T\mathbf{S}_1^T. \qquad (19)$$

**Update S**

Similarly, we can get the gradients w.r.t. $\mathbf{S}^L_i$ and $\mathbf{S}^R_i$ based on the chain rule,

$$\nabla_{\mathbf{S}^L_i}\mathcal{O}=\mathbf{F}_i^T\left[\nabla_{\mathbf{F}_{i-1}}\mathcal{O}\odot g^{'}(\mathbf{F}_i\mathbf{S}^L_i)\right], \qquad (20)$$

$$\nabla_{\mathbf{S}^R_i}\mathcal{O}=\left[\nabla_{\mathbf{G}_{i-1}}\mathcal{O}\odot g^{'}(\mathbf{S}^R_i\mathbf{G}_i)\right]\mathbf{G}_i^T, \qquad (21)$$

$$\nabla_{\mathbf{S}_1}\mathcal{O}=-\mathbf{F}_1^T(\mathbf{X}-\mathbf{F}_1\mathbf{S}_1\mathbf{G}_1-\mathbf{W}\mathbf{G}_{m_2})\mathbf{G}_1^T. \qquad (22)$$

**Update G**

The gradient w.r.t. $\mathbf{G}_i$ based on the chain rule is obtained as,

$$\begin{aligned}\nabla_{\mathbf{G}_1}\mathcal{O}&=-(\mathbf{F}_1\mathbf{S}_1)^T(\mathbf{X}-\mathbf{F}_1\mathbf{S}_1\mathbf{G}_1-\mathbf{W}\mathbf{G}_{m_2})\\ &\quad-\alpha\mathbf{P}_1^T(\mathbf{Y}-\mathbf{U}\mathbf{G}_1-\mathbf{V}\mathbf{G}_{m_2})\mathbf{Q}\mathbf{Q}^T.\\ \nabla_{\mathbf{G}_i}\mathcal{O}&=(\mathbf{S}^R_i)^T\left[\nabla_{\mathbf{G}_{i-1}}\mathcal{O}\odot g^{'}(\mathbf{S}^R_i\mathbf{G}_i)\right],1<i<m_2.\\ \nabla_{\mathbf{G}_{m_2}}\mathcal{O}&=(\mathbf{S}^R_{m_2})^T\left[\nabla_{\mathbf{G}_{m_2-1}}\mathcal{O}\odot g^{'}(\mathbf{S}^R_{m_2}\mathbf{G}_{m_2})\right]\\ &\quad-\mathbf{W}^T(\mathbf{X}-\mathbf{F}_1\mathbf{S}_1\mathbf{G}_1-\mathbf{W}\mathbf{G}_{m_2})\\ &\quad-\alpha\mathbf{P}^T(\mathbf{Y}-\mathbf{U}\mathbf{G}_1-\mathbf{V}\mathbf{G}_{m_2})\mathbf{Q}\mathbf{Q}^T.\end{aligned} \qquad (23)$$

These gradients on updating $\mathbf{F}$, $\mathbf{S}$ and $\mathbf{G}$ are propagated backwards in the multi-layer model, and in a specific layer, gradient descent is used to update the layer weights once gradients from the previous layer are received.

| Datasets | #Samples | #Dimensions | #Classes |
|---|---|---|---|
| ARface | 3120 | 560 | 120 |
| CMUPIE | 2856 | 1024 | 68 |
| Yale | 165 | 1024 | 15 |
| Caltech101-20 | 2386 | 928 | 20 |
| COIL20 | 1440 | 1024 | 20 |

Table 1: Statistic of datasets used in experiments. Here #Samples, #Dimensions and #Classes denote the number of samples, dimensions and classes, respectively.

### Update W, V and U

We derive the multiplicative update rules as follows:

$$\mathbf{W} = \mathbf{W} \odot \frac{\mathbf{X}\mathbf{G}_{m_2}^T}{(\mathbf{F}_1\mathbf{S}_1\mathbf{G}_1 + \mathbf{W}\mathbf{G}_{m_2})\mathbf{G}_{m_2}^T}, \tag{24}$$

$$\mathbf{U} = \mathbf{U} \odot \frac{\alpha\mathbf{Y}\mathbf{Q}\mathbf{Q}^T\mathbf{G}_1^T}{\alpha(\mathbf{U}\mathbf{G}_1 + \mathbf{V}\mathbf{G}_{m_2})\mathbf{Q}\mathbf{Q}^T\mathbf{G}_1^T}, \tag{25}$$

$$\mathbf{V} = \mathbf{V} \odot \frac{\alpha\mathbf{Y}\mathbf{Q}\mathbf{Q}^T\mathbf{G}_{m_2}^T}{\alpha(\mathbf{U}\mathbf{G}_1 + \mathbf{V}\mathbf{G}_{m_2})\mathbf{Q}\mathbf{Q}^T\mathbf{G}_{m_2}^T}. \tag{26}$$

## 5.3 Analysis on Computational Complexity

The detailed optimization, pseudocode and code of GD$^2$NMF are provided in the supplement[2]. Let $k_1$ and $k_2$ denote the maximum latent dimensions of basis matrices and feature matrices, respectively. For pre-training, the computational complexities of basis decomposition and feature decomposition are $O(m_1(k_1k_2p + k_1^2p))$ and $O(m_2(k_1k_2n + k_2^2n))$, respectively. For fine-tune, the bottleneck is updating $\mathbf{F}$, $\mathbf{S}$ and $\mathbf{G}$, which needs $O(m_1k_2pn)$, $O((m_1+m_2)k_1pn)$ and $O(m_2k_2n(c+k_2))$, respectively. Thus, the total time complexity of the algorithm is linear w.r.t. the data size.

# 6 Experiment

## 6.1 Experimental Settings

### Datasets

Five real-world datasets, including ARface[3], CMUPIE[4], Yale[5], Caltech101-20[6] and COIL20[7], are used in experiments. The statistics of used datasets are reported in Table 1, and more details are given in the supplement.

### Compared Methods

For *unsupervised* experiments, we compare GDNMF with six unsupervised NMF-based methods, including NMF [Lee and Seung, 1999], semi-NMF [Ding *et al.*, 2008], GNMF [Cai *et al.*, 2010], NeNMF [Guan *et al.*, 2012], DSNMF [Trigeorgis *et al.*, 2016] and DNBMF [Zhao *et al.*, 2021]. For *semi-supervised* experiments, we compare GD$^2$NMF with five semi-supervised NMF-based methods, including CNMF [Liu *et al.*, 2011], DNMF [Babaee *et al.*, 2016], Deep WSF [Trigeorgis *et al.*, 2016], DENMF [Wu *et al.*, 2019] and SGDNMF [Meng *et al.*, 2019].

### Configuration

In GDNMF and GD$^2$NMF, the decay rate $r$, which is the ratio of dimensions between successive layers, is selected from $\{0.3, \cdots, 0.7\}$, and the non-linear function $g(\cdot)$ is set to $g(x) = x^2$. The value of $\alpha$ used in GD$^2$NMF is selected from $\{10^{-3}, 10^{-2}, \cdots, 10^2\}$. For deep NMF methods, the number of layers is set to $m = 2$. Non-linear function $g(\cdot)$ in DSNMF and Deep WSF is set to $g(x) = 1.7159tanh(0.667x)$ as recommended by the authors. The value of $p$ used in GNMF is selected from $\{2, 3, ..., 9\}$. Other hyperparameters used in the compared methods are set according to the recommendation of the original papers. For semi-supervised methods, in experiments we randomly select 10% data points as labeled data. To evaluate the clustering results, after obtaining the latent representation $\mathbf{H}$, we run $k$-means on $\mathbf{H}$ for ten times with different initializations and calculate the mean results. In the experiment, three evaluation metrics, including cluster accuracy (ACC) [Xu *et al.*, 2003], normalized mutual information (NMI) [Cai *et al.*, 2008] and purity [Marutho *et al.*, 2018] are used to measure the clustering performance. ACC computes the percentage of correctly predicted cluster labels, NMI is used in clustering applications to measure the similarity of two clusters, and Purity measures how well classes are distributed on various clusters.

## 6.2 Experimental Results

### Evaluation on Clustering Performance

Table 2 shows the clustering results of unsupervised methods and semi-supervised methods on five datasets, where the best and the second-best results are highlighted in boldface and underlined, respectively. From Table 2, we can see that for *unsupervised* methods, GDNMF outperforms other comparing methods in most cases. Specifically, on the CMUPIE dataset, compared with the second-best method, GDNMF achieves performance improvement by 9.63%, 8.85% and 18.83% in terms of ACC, NMI and Purity, respectively. On the ARface dataset, DSNMF achieves the best results and GDNMF performs the second-best. For *semi-supervised* methods, GD$^2$NMF outperforms other comparing methods in most cases. On the CMUPIE dataset, compared with the second-best method (Deep WSF), GD$^2$NMF achieves performance improvement by 9.24%, 7.7% and 11.65% in terms of ACC, NMI and Purity, respectively. Moreover, GD$^2$NMF usually outperforms GDNMF, which validates the effectiveness of using the limited supervised information. The probable explanations for the performance superiority of GDNMF and GD$^2$NMF can be summarized as follows: 1) They perform deep basis and feature decomposition simultaneously, which helps to learn a hierarchical data representation based on multi-level basis. 2) Both methods incorporate shallow model and deep model in a unified architecture, which can capture both linear and nonlinear information. 3) GD$^2$NMF further improves the performance by utilizing prior infor-

[2] https://github.com/Gabrielx0098/GD2NMF

[3] https://www2.ece.ohio-state.edu/ aleix/ARdatabase.html

[4] https://www.cs.cmu.edu/afs/cs/project/PIE/MultiPie/Multi-Pie/Home.html

[5] http://cvc.cs.yale.edu/cvc/projects/yalefaces/yalefaces.html

[6] http://www.vision.caltech.edu/Image Datasets/Caltech101/

[7] https://www.cs.columbia.edu/CAVE/software/softlib/coil-20.php

| | | Unsupervised methods | | | | | | | Semi-supervised methods | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Data | Metric | NMF | semi-NMF | GNMF | NeNMF | DSNMF | DNBMF | **GDNMF** | CNMF | DNMF | Deep WSF | DENMF | SGDNMF | **GD²NMF** |
| COIL20 | ACC | 0.6111 | 0.5882 | **0.7422** | 0.6840 | 0.3694 | 0.7375 | 0.6313 | 0.6875 | 0.7181 | 0.3667 | 0.7243 | **0.8472** | 0.7535 |
| | NMI | 0.7456 | 0.7181 | **0.8100** | 0.7680 | 0.4326 | 0.8092 | 0.7481 | 0.7766 | 0.8143 | 0.3971 | 0.7635 | **0.8919** | 0.8173 |
| | Purity | 0.6077 | 0.5724 | 0.6482 | 0.6520 | 0.3169 | 0.6459 | **0.6521** | 0.6725 | 0.6672 | 0.3296 | 0.6639 | **0.7952** | 0.6701 |
| Arface | ACC | 0.4212 | 0.2907 | 0.3647 | 0.4130 | **0.4715** | 0.3526 | 0.4609 | 0.4670 | 0.3667 | 0.4732 | 0.0099 | 0.4664 | **0.4828** |
| | NMI | 0.6788 | 0.5340 | 0.6446 | 0.6937 | **0.7163** | 0.6331 | 0.6955 | 0.7048 | 0.6505 | 0.7135 | 0.1690 | 0.7101 | **0.7178** |
| | Purity | 0.4186 | 0.2892 | 0.3621 | 0.4228 | **0.5027** | 0.3468 | 0.4753 | 0.4394 | 0.3390 | **0.4909** | 0.0465 | 0.4619 | 0.4512 |
| CMUPIE | ACC | 0.7262 | 0.2875 | 0.5700 | 0.6618 | 0.7104 | 0.4688 | **0.8225** | 0.6432 | 0.4587 | 0.7661 | 0.2945 | 0.6849 | **0.8585** |
| | NMI | 0.8269 | 0.4412 | 0.7426 | 0.7843 | 0.8726 | 0.6681 | **0.9611** | 0.7835 | 0.6911 | 0.8814 | 0.5260 | 0.8610 | **0.9584** |
| | Purity | 0.6962 | 0.3061 | 0.5710 | 0.6293 | 0.7084 | 0.4640 | **0.8967** | 0.6524 | 0.4758 | 0.7202 | 0.3084 | 0.6686 | **0.8367** |
| Yale | ACC | 0.3879 | **0.4485** | 0.4364 | 0.3576 | 0.2242 | 0.4303 | **0.4485** | 0.5052 | 0.4727 | 0.2667 | 0.5073 | **0.5091** | **0.5091** |
| | NMI | 0.4472 | 0.4786 | 0.4808 | 0.4088 | 0.2536 | 0.4873 | **0.4886** | 0.5271 | 0.5210 | 0.3304 | 0.5454 | 0.5360 | **0.5530** |
| | Purity | 0.3891 | 0.4642 | 0.4055 | 0.3897 | 0.2370 | 0.4079 | **0.4667** | 0.4855 | 0.4194 | 0.2358 | 0.4964 | 0.4521 | **0.5091** |
| Caltech101-20 | ACC | 0.3894 | 0.4237 | 0.4324 | 0.4003 | 0.1911 | 0.3864 | **0.4349** | 0.4246 | 0.4158 | 0.2334 | 0.4590 | 0.4525 | **0.4655** |
| | NMI | 0.4866 | 0.5083 | 0.4751 | **0.5117** | 0.1559 | 0.4675 | 0.4460 | 0.5036 | 0.4736 | 0.1909 | 0.5165 | **0.5275** | 0.5215 |
| | Purity | 0.6596 | 0.6692 | 0.6478 | 0.6111 | 0.3909 | 0.6527 | **0.6746** | 0.6795 | 0.6698 | 0.3906 | 0.6886 | 0.6902 | **0.6954** |

Table 2: Clustering performance of comparing methods on five real-world datasets. The best results are highlighted in boldface, while the second-best results are underlined.

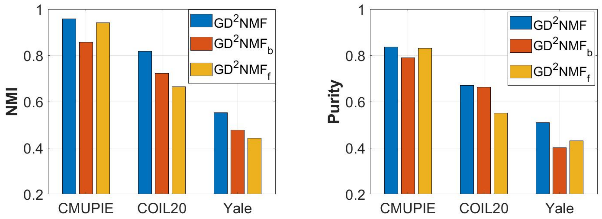

Figure 2: Comparison of GD²NMF, GD²NMF$_b$ and GD²NMF$_f$ on three datasets. GD²NMF$_b$ and GD²NMF$_f$ only consider basis decomposition and feature decomposition, respectively.

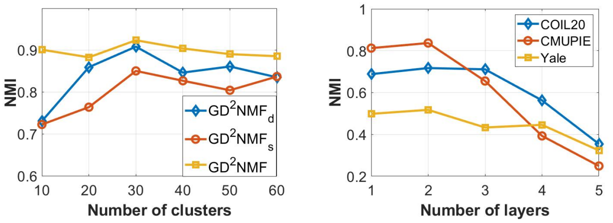

Figure 3: Case study on the structure of GD²NMF. GD²NMF$_d$ and GD²NMF$_s$ are variants of GD²NMF by ignoring shallow and deep models, respectively. The right subfigure reports the performance of GD²NMF by varing the number of layers from 1 to 5 by step 1.

mation in a discriminative manner. However, GDNMF and GD²NMF didn't achieve the best results on ARface and COIL20, probably because the deep model is too complex for these two datasets, and more efforts should be paid for hyperparameter tuning.

### Ablation Study[8]

To evaluate the effectiveness of deep feature and basis decomposition in GD²NMF, we conduct an experiment to compare the clustering results of GD²NMF with its two degraded variants: GD²NMF with only basis decomposition (GD²NMF$_b$) and GD²NMF with only feature decomposition (GD²NMF$_f$). The experiment is conducted on the CMUPIE, Yale and COIL20 datasets, and results are shown in Fig.2. Similar results are observed on other datasets. For fair comparison, the hyperparameters are set to the same values. From Fig.2, we can see that GD²NMF achieves better performance than both GD²NMF$_b$ and GD²NMF$_f$ on the three datasets, which validates our assumption that simultaneously performing deep factorization on features and bases leads to improved data representation and clustering performance.

To verify the effectiveness of incorporating deep and shallow models in GD²NMF, we conducted an experiment on CMUPIE to compare the performance of GD²NMF, GD²NMF in shallow architecture (GD²NMF$_s$) and GD²NMF in deep architecture (GD²NMF$_d$). In the experiment, we extract different number of classes from the original CMUPIE dataset, and generate six datasets by varying the number of classes from 20 to 60 by step 10. Results in NMI are shown in the left subfigure of Fig.3. Obviously, by combining shallow and deep models in a single architecture, GD²NMF consistently achieves a better performance than its two variants, which demonstrates that mutually reinforcing shallow and deep models indeed helps to boost the generalization performance. In addition, we conduct an experiment to evaluate the performance of GD²NMF by varying the number of layers from 1 to 5, and report the results in the right subfigure of Fig.3. We can see that a two-layer structure leads to the best performance. When the number of layers is greater than 2, its performance drops significantly, probably because the structure is too complex and easy to overfit.

## 7 Conclusion

This paper proposes a novel deep NMF method, named GDNMF, which learns a hierarchical data representation based on multi-level basis by simultaneously performing deep factorization on the feature matrix and the basis matrix. GDNMF incorporates linear shallow model and non-linear deep model in a unified architecture, that generalizes several existing NMF-based methods. Moreover, semi-supervised GDNMF is proposed to utilize partial label information in a discriminative way. Experiments on five real-world datasets show the effectiveness of GDNMF.

[8] More experimental results are provided in the supplement.

## References

[Babaee *et al.*, 2016] Mohammadreza Babaee, Stefanos Tsoukalas, Maryam Babaee, Gerhard Rigoll, and Mihai Datcu. Discriminative nonnegative matrix factorization for dimensionality reduction. *Neurocomputing*, 173:212–223, 2016.

[Baraniuk, 2011] Richard G Baraniuk. More is less: Signal processing and the data deluge. *Science*, 331(6018):717–719, 2011.

[Buckler *et al.*, 2018] Mark Buckler, Philip Bedoukian, Suren Jayasuriya, and Adrian Sampson. Eva$^2$: Exploiting temporal redundancy in live computer vision. In *2018 ACM/IEEE 45th Annual International Symposium on Computer Architecture (ISCA)*, pages 533–546. IEEE, 2018.

[Cai *et al.*, 2008] Deng Cai, Xiaofei He, Xiaoyun Wu, and Jiawei Han. Non-negative matrix factorization on manifold. In *2008 eighth IEEE international conference on data mining*, pages 63–72. IEEE, 2008.

[Cai *et al.*, 2010] Deng Cai, Xiaofei He, Jiawei Han, and Thomas S Huang. Graph regularized nonnegative matrix factorization for data representation. *IEEE transactions on pattern analysis and machine intelligence*, 33(8):1548–1560, 2010.

[Chen *et al.*, 2019] Zhensong Chen, Yong Shi, and Zhiquan Qi. Constrained matrix factorization for semi-weakly learning with label proportions. *Pattern Recognition*, 91:13–24, 2019.

[Chen *et al.*, 2022] Wen-Sheng Chen, Qianwen Zeng, and Binbin Pan. A survey of deep nonnegative matrix factorization. *Neurocomputing*, 491:305–320, 2022.

[Ding *et al.*, 2008] Chris HQ Ding, Tao Li, and Michael I Jordan. Convex and semi-nonnegative matrix factorizations. *IEEE transactions on pattern analysis and machine intelligence*, 32(1):45–55, 2008.

[Fang *et al.*, 2018] Hao Fang, Aihua Li, Tao Wang, Hongwei Chang, and Huoxi Xu. Hyperspectral unmixing using graph-regularized and sparsity-constrained deep nmf. In *Optical Sensing and Imaging Technologies and Applications*, volume 10846, pages 664–672. SPIE, 2018.

[Guan *et al.*, 2012] Naiyang Guan, Dacheng Tao, Zhigang Luo, and Bo Yuan. Nenmf: An optimal gradient method for nonnegative matrix factorization. *IEEE Transactions on Signal Processing*, 60(6):2882–2898, 2012.

[He and Niyogi, 2003] Xiaofei He and Partha Niyogi. Locality preserving projections. *Advances in neural information processing systems*, 16, 2003.

[Karczmarek *et al.*, 2019] P Karczmarek, W Pedrycz, A Kiersztyn, and M Dolecki. A comprehensive experimental comparison of the aggregation techniques for face recognition. *Iranian Journal of Fuzzy Systems*, 16(4):1–19, 2019.

[Lan *et al.*, 2014] Long Lan, Naiyang Guan, Xiang Zhang, Dacheng Tao, and Zhigang Luo. Soft-constrained nonnegative matrix factorization via normalization. In *2014 International Joint Conference on Neural Networks (IJCNN)*, pages 3025–3030. IEEE, 2014.

[Lee and Seung, 1999] Daniel D Lee and H Sebastian Seung. Learning the parts of objects by non-negative matrix factorization. *Nature*, 401(6755):788–791, 1999.

[Li and Yuan, 2004] Ming Li and Baozong Yuan. A novel statistical linear discriminant analysis for image matrix: two-dimensional fisherfaces. In *Proceedings 7th International Conference on Signal Processing, 2004. Proceedings. ICSP'04. 2004.*, volume 2, pages 1419–1422. IEEE, 2004.

[Liu *et al.*, 2011] Haifeng Liu, Zhaohui Wu, Xuelong Li, Deng Cai, and Thomas S Huang. Constrained nonnegative matrix factorization for image representation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 34(7):1299–1311, 2011.

[Liu *et al.*, 2017] Yalin Liu, Naiyang Guan, and Jie Liu. Deep transductive nonnegative matrix factorization for speech separation. In *2017 16th IEEE International Conference on Machine Learning and Applications (ICMLA)*, pages 249–254. IEEE, 2017.

[Marutho *et al.*, 2018] Dhendra Marutho, Sunarna Hendra Handaka, Ekaprana Wijaya, et al. The determination of cluster number at k-mean using elbow method and purity evaluation on headline news. In *2018 international seminar on application for technology of information and communication*, pages 533–538. IEEE, 2018.

[Meng *et al.*, 2019] Yang Meng, Ronghua Shang, Fanhua Shang, Licheng Jiao, Shuyuan Yang, and Rustam Stolkin. Semi-supervised graph regularized deep nmf with biorthogonal constraints for data representation. *IEEE transactions on neural networks and learning systems*, 31(9):3245–3258, 2019.

[Moon and Lee, 2021] Sehwan Moon and Hyunju Lee. Jdsnmf: Joint deep semi-non-negative matrix factorization for learning integrative representation of molecular signals in alzheimer's disease. *Journal of personalized medicine*, 11(8):686, 2021.

[Moore, 1981] Bruce Moore. Principal component analysis in linear systems: Controllability, observability, and model reduction. *IEEE transactions on automatic control*, 26(1):17–32, 1981.

[Peng *et al.*, 2021a] Chong Peng, Zhilu Zhang, Zhao Kang, Chenglizhao Chen, and Qiang Cheng. Nonnegative matrix factorization with local similarity learning. *Information Sciences*, 562:325–346, 2021.

[Peng *et al.*, 2021b] Siyuan Peng, Wee Ser, Badong Chen, and Zhiping Lin. Robust semi-supervised nonnegative matrix factorization for image clustering. *Pattern Recognition*, 111:107683, 2021.

[Sabzalian and Abolghasemi, 2018] B Sabzalian and V Abolghasemi. Iterative weighted non-smooth nonnegative matrix factorization for face recognition. *International Journal of Engineering*, 31(10):1698–1707, 2018.

[Shi *et al.*, 2018] Tian Shi, Kyeongpil Kang, Jaegul Choo, and Chandan K Reddy. Short-text topic modeling via non-negative matrix factorization enriched with local word-context correlations. In *Proceedings of the 2018 World Wide Web Conference*, pages 1105–1114, 2018.

[Tong *et al.*, 2019] Ming Tong, Yiran Chen, Mengao Zhao, Haili Bu, and Shengnan Xi. A deep discriminative and robust nonnegative matrix factorization network method with soft label constraint. *Neural Computing and Applications*, 31(11):7447–7475, 2019.

[Trigeorgis *et al.*, 2016] George Trigeorgis, Konstantinos Bousmalis, Stefanos Zafeiriou, and Björn W Schuller. A deep matrix factorization method for learning attribute representations. *IEEE transactions on pattern analysis and machine intelligence*, 39(3):417–429, 2016.

[Wang *et al.*, 2021] Wei Wang, Feiyu Chen, Yongxin Ge, Sheng Huang, Xiaohong Zhang, and Dan Yang. Discriminative deep semi-nonnegative matrix factorization network with similarity maximization for unsupervised feature learning. *Pattern Recognition Letters*, 149:157–163, 2021.

[Wisdom *et al.*, 2017] Scott Wisdom, Thomas Powers, James Pitton, and Les Atlas. Deep recurrent nmf for speech separation by unfolding iterative thresholding. In *2017 IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA)*, pages 254–258. IEEE, 2017.

[Wu *et al.*, 2019] Wenhui Wu, Sam Kwong, Junhui Hou, Yuheng Jia, and Horace Ho Shing Ip. Simultaneous dimensionality reduction and classification via dual embedding regularized nonnegative matrix factorization. *IEEE Transactions on Image Processing*, 28(8):3836–3847, 2019.

[Xu *et al.*, 2003] Wei Xu, Xin Liu, and Yihong Gong. Document clustering based on non-negative matrix factorization. In *Proceedings of the 26th annual international ACM SIGIR conference on Research and development in informaion retrieval*, pages 267–273, 2003.

[Yi *et al.*, 2019] Yugen Yi, Jianzhong Wang, Wei Zhou, Caixia Zheng, Jun Kong, and Shaojie Qiao. Non-negative matrix factorization with locality constrained adaptive graph. *IEEE Transactions on circuits and systems for video technology*, 30(2):427–441, 2019.

[Zhao *et al.*, 2021] Yang Zhao, Huiyang Wang, and Jihong Pei. Deep non-negative matrix factorization architecture based on underlying basis images learning. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 43(6):1897–1913, 2021.

[Zhao *et al.*, 2022] Yang Zhao, Furong Deng, Jihong Pei, and Xuan Yang. Progressive deep non-negative matrix factorization architecture with graph convolution-based basis image reorganization. *Pattern Recognition*, 132:108984, 2022.