Nonlinear Non-negative Matrix Factorization using Deep Learning

Hui Zhang[1,3], Huaping Liu[2,3*], Rui Song[1], Fuchun Sun[2,3]
[1] School of Control Science and Engineering, Shandong University, Jinan, China
[2] Department of Computer Science and Technology, Tsinghua University, Beijing, China
[3] State Key Lab. of Intelligent Technology and Systems, Tsinghua University, TNLIST, Beijing, China
[*]Email: hpliu@tsinghua.edu.cn

Abstract—**In this paper, we describe the deep learning method to reduce the dimension of the data samples under the framework Non-negative Matrix Factorization (NMF). That is to say, we try to find the good representation of the data samples for the task of NMF. To this end, a nonlinear NMF optimization model is constructed and the optimization algorithm is developed. The experimental results on some benchmark dataset show the nonlinear dimension reduction helps the NMF to improve the clustering performance.**

Keywords—deep learning; nonlinear dimensional reduction; non-negative matrix factorization

I. INTRODUCTION

Non-negative Matrix Factorization (NMF), which aims to factorize a matrix into two non-negative matrices whose product reconstructs the original data matrix, has been shown extensive applications in many domains such as signal processing [1], machine learning [2], text mining [3], and so on. It has been found that such a factorization exhibits many favorable properties such as sparsity and interpretability. In addition, the obtained part-based representation is consistent to the psychological and physiological evidence in human brain [4].

Despite the great success of NMF, there exists an intrinsic problem. The basic NMF model suggests that signals can be efficiently approximated as a linear combination of dictionary atoms. However, the linear reconstruction assumption which is explicitly imposed by many existing work is not valid for many complicated signals. In [5], the authors constructed an affinity graph to encode the geometrical information and seek a matrix factorization which respected the graph structure. Such a model provided more powerful representation capability.

On the other hand, by exploiting the "kernel trick", input samples are implicitly mapped to a higher dimensional Reproducing Kernel Hilbert Space. This is useful in dealing with nonlinear data as the form of feature transformation can be readily controlled by using different types of kernel functions. In [6], kernel NMF was originally proposed. The calculation of the kernel NMF does not need to know the actual value of observations, and only the kernel matrix is required. The nonlinear mapping enables the data to represent their correlation in the high dimensional space. Three significant advantages of kernel NMF over NMF are discovered including extracting more useful features hidden in the original data, solving the problem of the data where only relationships between objects are known, and being able to process the data with negative values by using some specific kernel functions [7].

However, even if the kernel trick can be used to address the nonlinearity issue, these methods still suffer from the scalability problem because they cannot obtain the explicit nonlinear mapping function. In addition, how the select the kernel function is a difficult problem.

Recently, the deep learning methodology has attracted much attentions since an efficient lay-wise unsupervised learning strategy can be used to pre-train the deep architectures. It shows that deep learning architectures could provide powerful representability and achieved great success in face recognition [8], digit recognition [9], and clustering [10]. This motivates us to solve the NMF problem in the new feature space.

In this paper, we borrow the idea of deep learning and propose a nonlinear NMF learning method to solve the clustering problem. The main contributions are summarized as follows:

(1) We develop the deep learning method to tackle the nonlinear NMF problem.

(2) An optimization algorithm is developed to solve the presented nonlinear NMF problem

(3) Extensive experimental results shows that the proposed method exhibits superior advantages.

This paper is organized as follows: In Section 2 we discuss the relation of our work to prior work. Section 3 presents the brief review about NMF method. In Sections 4-5 we formulate the problem and develop the optimization algorithm. Section 6 shows some experimental results.

II. RELATION TO PROPR WORK

There exists some work which combines the deep learning method and NMF. In this section, we point the difference between our model and existing work.

Most of the existing work regarded the deep learning and NMF as two separate modules. For example, in the system developed by [11], the sparse NMF was used to extract features from the noisy observation, followed by a deep neural network for classification. The work in [12] combined separate modules of deep learning and NMF for speech recognition. In [13], the proposed approach used two stages of deep neural networks, where the first stage estimated the ideal ratio mask that separated

978-1-5090-0620-5/16/$31.00

speech from noise, and the second stage mapped the ratio-masked speech to the clean speech activation matrices that were used for NMF. In [14], the authors proposed an approach to improve encoding vector estimation for target signal extraction. The estimating encoding vectors from the mixture data is viewed as a regression problem and a deep neural network is used to learn the mapping between the mixture data and the corresponding encoding vectors. In addition, in [15], NMF was used to initialize the DNN estimate for each source.

It should be noted that in [16] [17], the Deep NMF model was proposed. However, the core contributor of the so-called deep NMF is to factorize the original data matrix into many matrix and obtain the hierarchical representation.

To summarize, none of the above work addresses the problem of simultaneous dimensional reduction on and NMF. Our work tries to solve this problem by resort to the deep learning method to obtain the nonlinear dimension reduction for the NMF. To the best of our knowledge, this problem has never been addressed.

III. Brief review of NMF

In this section we give a brief review about the NMF method. NMF aims to discover two non-negative matrices $\boldsymbol{D} \in R^{n\times k}$ and $\boldsymbol{X} \in R^{k\times m}$ to minimize the following representation error

$$E = \|\boldsymbol{V} - \boldsymbol{D}\boldsymbol{X}\|_F^2 \qquad s.t. \;\; \boldsymbol{D}, \boldsymbol{X} \geq 0 \tag{1}$$

where $\boldsymbol{V} = [\boldsymbol{v}^1, \boldsymbol{v}^2, \cdots, \boldsymbol{v}^m] \in \mathrm{R}^{n\times m}$ represents the original data samples, each column of $\boldsymbol{V}$ is a sample vector, $\|\cdot\|_F$ denotes the matrix Frobenius norm, $\boldsymbol{D}$ and $\boldsymbol{X}$ are two non-negative matrices. The objective function is not convex in both variables together. Lee and Seung[4] proposed an algorithm to optimize the problem in (1) by updating $\boldsymbol{D}$ and $\boldsymbol{X}$ iteratively as follows:

$$\boldsymbol{D}_{ij} \leftarrow \boldsymbol{D}_{ij} \frac{(\boldsymbol{V}\boldsymbol{X}^T)_{ij}}{(\boldsymbol{D}\boldsymbol{X}\boldsymbol{X}^T)_{ij}} \tag{2}$$

$$\boldsymbol{X}_{ij} \leftarrow \boldsymbol{X}_{ij} \frac{(\boldsymbol{D}^T\boldsymbol{V})_{ij}}{(\boldsymbol{D}^T\boldsymbol{D}\boldsymbol{X})_{ij}} \tag{3}$$

It has been proved in [4] that the objective function in (1) is non-increasing under the above update rules, and invariant if and only if $\boldsymbol{D}$ and $\boldsymbol{X}$ are at a stationary point of the distance.

Since NMF only models the linear reconstruction relation, it performs poorly on nonlinear structured data. To this end, Ref.[5] proposed Graph regularized Non-negative Matrix Factorization (GNMF) to models the data space as a sub-manifold in the ambient space and performs the NMF on this manifold.

According to the idea of GNMF, a graph has n vertices, and each vertex corresponds to a data point. Define the edge weight matrix $\boldsymbol{M}$ as follows:

$$\boldsymbol{M}_{ij} = \begin{cases} 1, & if\ \boldsymbol{v}^i \epsilon N_p(\boldsymbol{v}^j)\ or\ \boldsymbol{v}^j \epsilon N_p(\boldsymbol{v}^i) \\ 0, & otherwise. \end{cases} \tag{4}$$

where $N_p(\boldsymbol{v}^j)$ denotes the p nearest neighbors of $\boldsymbol{v}^j$. Define the graph Laplacian $\boldsymbol{L} = \boldsymbol{Q} - \boldsymbol{M}$ where $\boldsymbol{Q}$ is a diagonal matrix and $\boldsymbol{Q}_{ii} = \sum_{j=1}^{m} \boldsymbol{M}_{ij}$. The GNMF incorporates the graph regularization term with NMF and minimizes the objective function

$$\tilde{E} = \|\boldsymbol{V} - \boldsymbol{D}\boldsymbol{X}\|_F^2 + \alpha\, Tr(\boldsymbol{X}\boldsymbol{L}\boldsymbol{X}^T) \quad s.t. \boldsymbol{D}, \boldsymbol{X} \geq 0 \tag{5}$$

where α denotes the weight of regularized item, and $Tr(\cdot)$ denotes the trace of a matrix. The new rules to update the factors $\boldsymbol{D}$ and $\boldsymbol{X}$ is

$$\boldsymbol{D}_{ij} \leftarrow \boldsymbol{D}_{ij} \frac{(\boldsymbol{V}\boldsymbol{X}^T)_{ij}}{(\boldsymbol{D}\boldsymbol{X}\boldsymbol{X}^T)_{ij}} \tag{6}$$

$$\boldsymbol{X}_{ij} \leftarrow \boldsymbol{X}_{ij} \frac{(\boldsymbol{D}^T\boldsymbol{V} + \alpha\boldsymbol{X}\boldsymbol{M})_{ij}}{(\boldsymbol{D}^T\boldsymbol{D}\boldsymbol{X} + \alpha\boldsymbol{X}\boldsymbol{Q})_{ij}} \tag{7}$$

It is also proved that the above update steps can be applied to find local minima of the objective function in (5).

IV. Problem formulation

Classical NMF and GNMF decompose raw data matrix into two non-negative matrices. Our method utilizes the deep neural network to achieve Nonlinear Non-negative Matrix Factorization (N-NMF) and Nonlinear Graph regularized Non-negative Matrix Factorization (N-GNMF), which can discover nonlinear geometrical feature of the data space.

A. Nonlinear Non-negative Matrix Factorization (N-NMF)

Different from the classical NMF, we propose a new algorithm that transforms the raw sample to a low-dimensional feature space nonlinearly, and then decomposes the output signals with two non-negative matrixes. The nonlinear mapping is denoted as a nonlinear function $f(\cdot): \mathrm{R}^n \rightarrow \mathrm{R}^d$ that transforms a sample $\boldsymbol{v}$ to a d-dimensional output. We learn the nonlinear mapping together with the non-negative matrix factorization to minimize the N-NMF error:

$$\min_{\boldsymbol{f},\boldsymbol{D},\boldsymbol{X}} \boldsymbol{E}_{N-NMF} = \|\boldsymbol{f}(\boldsymbol{V}) - \boldsymbol{D}\boldsymbol{X}\|_F^2 \quad s.t. \boldsymbol{f}(\boldsymbol{V}), \boldsymbol{D}, \boldsymbol{X} \geq 0 \tag{8}$$

Deep neural network [18] is very popular in these years, which can extract complex hidden features from the raw data. Our mathod utilizes deep neural network to realize the nonlinear mapping. Fig. 1 is an example of the deep neural network conveying the raw data to the feature space. The output feature of an input $\boldsymbol{v}$ is

$$f(\boldsymbol{v}) = \sigma(\boldsymbol{W}_3\sigma(\boldsymbol{W}_2\sigma(\boldsymbol{W}_1\boldsymbol{v} + \boldsymbol{b}_1) + \boldsymbol{b}_2) + \boldsymbol{b}_3) \tag{9}$$

where $\sigma(x) = 1/(1 + e^{-x})$ is the sigmoid function, $\{\boldsymbol{W}_k\}_{k=1,2,3}$ and $\{\boldsymbol{b}_k\}_{k=1,2,3}$ are the weights and bias terms of this deep neural network. The non-negative constraint on the output $f(\boldsymbol{v})$ is guaranteed as the output of sigmoid function is non-negative.

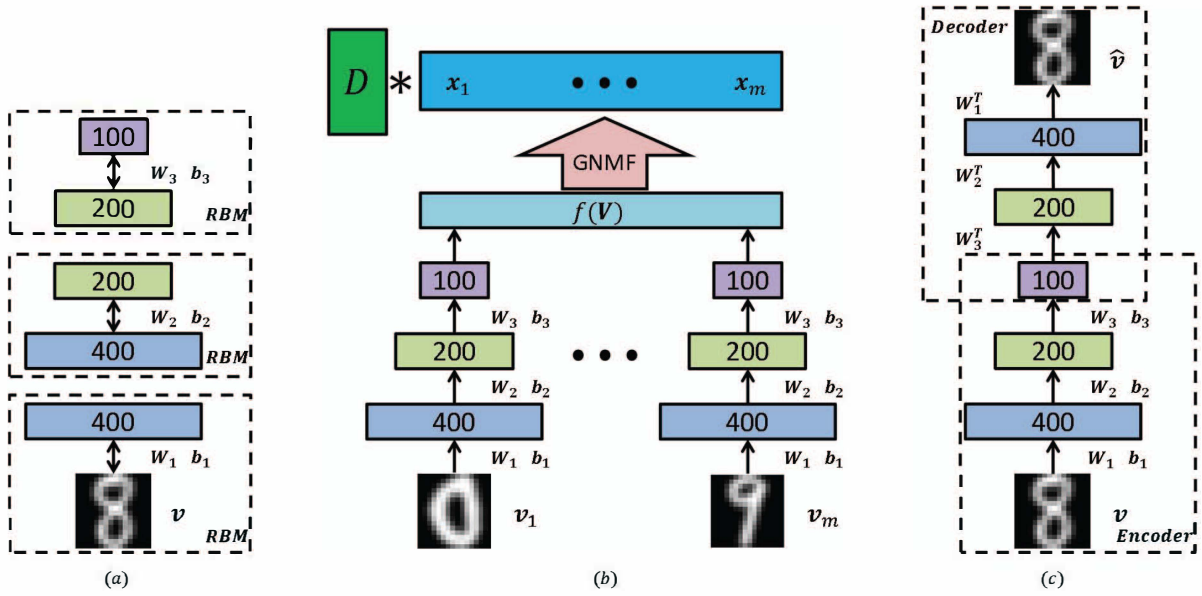

Fig. 1. (a) Pretraining (b) Nonlinear GNMF based on Deep Network (c) Deep Autoencode

Only using this constraint of (8) is not enough to learn an optimal mapping extracting effective feature. We hope the output feature $f(\boldsymbol{V})$ preserve the useful infromation in raw samples. Unrolling the encoder network as shown in Fig.1(c), we can get a decoder network and creat a deep autoencoder. The decoder weights are tied with encoder weights, but drop the bias terms which have no effects to the output feature $f(\boldsymbol{V})$. Decoder network works as a transformation $\hat{f}(\cdot)$: $\mathrm{R}^d \rightarrow \mathrm{R}^n$ which reconstruct the original sample $\boldsymbol{v}$ using the transformed feature $f(\boldsymbol{v})$, as the following representation:

$$\hat{f}(\boldsymbol{v}) = \sigma\left(\boldsymbol{W}_1^T\sigma\left(\boldsymbol{W}_2^T\sigma(\boldsymbol{W}_3^T f(\boldsymbol{v}))\right)\right) \tag{10}$$

After obtaining the reconstructed value of $\hat{\boldsymbol{v}} = \hat{f}(\boldsymbol{v})$, we can use the reconstruction error to refine the learned feature effectively. We define reconstruction error of sample $\boldsymbol{v}^r$ as the cross entropy [9][10]:

$$\boldsymbol{Er}^r = -\sum_i v_i^r \log \hat{v}_i^r - \sum_i (1 - v_i^r)\log(1 - \hat{v}_i^r) \tag{11}$$

where $v_i^r \in [0,1]$ denotes the element i for sample $\boldsymbol{v}^r$, and $\hat{v}_i^r$ denotes its reconstruction.

Our cost function $J(\boldsymbol{W}, \boldsymbol{B}, \boldsymbol{D}, \boldsymbol{X})$ combines the constraint of N-NMF error and reconstruction error to minimize:

$$J(\boldsymbol{W}, \boldsymbol{B}, \boldsymbol{D}, \boldsymbol{X}) = \boldsymbol{E}_{N-NMF} + \lambda \sum_{r=1}^{m} \boldsymbol{Er}^r \tag{12}$$

where λ is the weight of reconstruction error.

B. Nonlinear Graph regularized Non-negative Matrix Factorization (N-GNMF)

It's easy to add graph regularization to N-NMF algorithm, and achieve Nonlinear Graph regularized Non-negative Matrix Factorization (N-GNMF). The cost function of N-GNMF changes $\boldsymbol{E}_{N-NMF}$ in (12) to $\boldsymbol{E}_{N-GNMF}$, which is defined as:

$$\boldsymbol{E}_{N-GNMF} = \|f(\boldsymbol{V}) - \boldsymbol{DX}\|_F^2 + \alpha\, Tr(\boldsymbol{XLX^T}) \tag{13}$$

where α denotes the weight of regularized item, and $Tr(\cdot)$ denotes the trace of a matrix.

V. Algorithm

In our algorithm, we perform two steps circularly to minimize the objective in (12). First, we update the parameters of the deep neural network via gradient decent to learn a better nonlinear mapping with fixed NMF factors $\boldsymbol{D}$ and $\boldsymbol{X}$. The derivatives of $\boldsymbol{E}_{N-NMF}$ with respect to parameter $\boldsymbol{W}$ are:

$$\frac{\partial \boldsymbol{E}_{N-NMF}}{\partial \boldsymbol{W}} = \frac{\partial f(V)}{\partial \boldsymbol{W}} \frac{\partial \boldsymbol{E}_{N-NMF}}{\partial f(V)} \tag{14}$$

where

$$\frac{\partial \boldsymbol{E}_{N-NMF}}{\partial f(V)} = 2[f(\boldsymbol{V}) - \boldsymbol{DX}]$$

and $\frac{\partial f(V)}{\partial \boldsymbol{W}}$ can be computed using standard backpropagation. The derivatives of $\boldsymbol{Er}^r$ with respect to parameter $\boldsymbol{W}$ are:

$$\frac{\partial \boldsymbol{Er}^r}{\partial \boldsymbol{W}} = \frac{\partial \hat{f}(\boldsymbol{v}^r)}{\partial \boldsymbol{W}} \frac{\partial \boldsymbol{Er}^r}{\partial \hat{f}(\boldsymbol{v}^r)} \tag{15}$$

where $\frac{\partial \boldsymbol{Er}^r}{\partial \hat{f}(\boldsymbol{v}^r)} \in \mathrm{R}^{n\times 1}$, and its i-th element is

$$\left\{\frac{\partial \boldsymbol{Er}^r}{\partial \hat{f}(\boldsymbol{v}^r)}\right\}_i = -\frac{v_i^r}{\hat{v}_i^r} + \frac{1 - v_i^r}{1 - \hat{v}_i^r}$$

Although the encoder weights and decoder weights are tied together, we assume them detached so that we can compute $\frac{\partial \hat{f}(\boldsymbol{v}^r)}{\partial \boldsymbol{W}}$ with standard backpropagation algorithm. The actual

derivative to every parameter is separated into two parts, and it's easy to sum them together after backpropagation. The derivatives of $\boldsymbol{E}_{N-NMF}$ and $\boldsymbol{Er}^{r}$ with respect to parameter $\boldsymbol{b}$ are similar, and the decoder network has no bias terms to be cared about.

Once the network is updated, a second step is performed to search for better factorization of NMF. The reconstruction error term in (12) is not influenced by the factors $\boldsymbol{D}$ and $\boldsymbol{X}$. The problem to search for better factorization is equivalent to minimize the N-NMF error $\boldsymbol{E}_{N-NMF}$. So we can utilize the algorithm proposed by Lee and Seung [4], updating $\boldsymbol{D}$ and $\boldsymbol{X}$ iteratively as follows:

$$\boldsymbol{D}_{ij} \leftarrow \boldsymbol{D}_{ij} \frac{(f(\boldsymbol{V})\boldsymbol{X}^{T})_{ij}}{(\boldsymbol{D}\boldsymbol{X}\boldsymbol{X}^{T})_{ij}} \tag{16}$$

$$\boldsymbol{X}_{ij} \leftarrow \boldsymbol{X}_{ij} \frac{\left(\boldsymbol{D}^{T} f(\boldsymbol{V})\right)_{ij}}{(\boldsymbol{D}^{T}\boldsymbol{D}\boldsymbol{X})_{ij}} \tag{17}$$

Before training the network and NMF factors, we should initialize them first. The parameters of network are initialized by training Restricted Boltzmann Machine layer-wise. An initial network is obtained after pretraining, expressed as $f_0(\cdot)$. Set factors $\boldsymbol{D}$ and $\boldsymbol{X}$ with non-negative random number, and repeat (16) and (17) for T-times to learn an initial NMF factors (T=100 in our experiments). A summary of the whole algorithm is described in Fig. 2. In practice, we empirically iterate to update the factors $\boldsymbol{D}$ and $\boldsymbol{X}$ 10 times in each echo.

Input: Cluster Samples: $\boldsymbol{V} = [\boldsymbol{v}^1, \boldsymbol{v}^2, \cdots, \boldsymbol{v}^m] \in \mathrm{R}^{n\times m}$

1 Initialization:

Pretrain $W^{(0)}, b^{(0)}$ using RBM.

Set non-negative $\boldsymbol{D}^{(0)} \in \boldsymbol{R}^{n\times k}, \boldsymbol{X}^{(0)} \in \boldsymbol{R}^{k\times m}$.

2 Repeat T-times

$$\boldsymbol{D}_{ij} \leftarrow \boldsymbol{D}_{ij} \frac{(f(\boldsymbol{V})\boldsymbol{X}^{T})_{ij}}{(\boldsymbol{D}\boldsymbol{X}\boldsymbol{X}^{T})_{ij}} \qquad \boldsymbol{X}_{ij} \leftarrow \boldsymbol{X}_{ij} \frac{\left(\boldsymbol{D}^{T} f(\boldsymbol{V})\right)_{ij}}{(\boldsymbol{D}^{T}\boldsymbol{D}\boldsymbol{X})_{ij}}$$

End

3 Repeat N-times

Repeat 10-times

$$\boldsymbol{D}_{ij} \leftarrow \boldsymbol{D}_{ij} \frac{(f(\boldsymbol{V})\boldsymbol{X}^{T})_{ij}}{(\boldsymbol{D}\boldsymbol{X}\boldsymbol{X}^{T})_{ij}} \qquad \boldsymbol{X}_{ij} \leftarrow \boldsymbol{X}_{ij} \frac{\left(\boldsymbol{D}^{T} f(\boldsymbol{V})\right)_{ij}}{(\boldsymbol{D}^{T}\boldsymbol{D}\boldsymbol{X})_{ij}}$$

End

$$\boldsymbol{W} := \boldsymbol{W} - \alpha \frac{\partial}{\partial \boldsymbol{W}} J(\boldsymbol{W}, \boldsymbol{b})$$

$$\boldsymbol{b} := \boldsymbol{b} - \alpha \frac{\partial}{\partial \boldsymbol{b}} J(\boldsymbol{W}, \boldsymbol{b})$$

End

Fig. 2. The Nonlinear-NMF Algorithm

N-GNMF algorithm is similar to N-NMF algorithm, except the graph regularized item which has no effects to the network training and only affects the factors $\boldsymbol{D}$ and $\boldsymbol{X}$. As for N-GNNF, the new rule to update the factors $\boldsymbol{D}$ and $\boldsymbol{X}$ is

$$\boldsymbol{D}_{ij} \leftarrow \boldsymbol{D}_{ij} \frac{(f(\boldsymbol{V})\boldsymbol{X}^{T})_{ij}}{(\boldsymbol{D}\boldsymbol{X}\boldsymbol{X}^{T})_{ij}} \tag{18}$$

$$\boldsymbol{X}_{ij} \leftarrow \boldsymbol{X}_{ij} \frac{(\boldsymbol{D}^{T} f(\boldsymbol{V}) + \alpha \boldsymbol{X}\boldsymbol{M})_{ij}}{(\boldsymbol{D}^{T}\boldsymbol{D}\boldsymbol{X} + \alpha \boldsymbol{X}\boldsymbol{Q})_{ij}} \tag{19}$$

In the GNMF constraint, there are two parameters: the nearest neighbors p and the graph regularization α. We set the number of nearest neighbors p to 5 and the value of regularization α to 100 as in [5].

VI. Experimental Results

In this section, we evaluate N-NMF and N-GNMF algorithm by comparing to three other approaches (GNMF, NMF and K-means) on six datasets (Yale-B, ORL, COIL20, COIL100, USPS, DIGITS).

A. Description of Datesets

A brief description of the six datasets used in the experiments is provided below:

Yale-B [19] and ORL [20] are both face image datasets. Yale-B contains 576 viewing conditions (9 poses × 64 illuminations) for 10 individuals. ORL contains 10 different images of 40 individuals. For computational consideration, Yale-B[1] is resized to 30×40 pixels, and ORL is rescaled into a size of 28×23 pixels.

COIL-20[2] and COIL-100[3] are both object datasets, containing 20 objects and 100 objects respectively. Each object contains 72 images shot in different angles. We use the processed version of the two datasets. For computational consideration, all images are rescaled into a size of 32×32 pixels.

USPS[21] and DIGITS[4] are both handwritten datasets, containing 10 digits from 0 to 9. For the USPS and DIGITS data sets, we randomly sample 500 data points from each of the 10 classes.

Each image of these datasets are conveyed into a vector in column-major order. The details of all databases are summarized in Table I.

TABLE I. Details of the Datasets Used in the Experiments

Dataset	Size	Dim	Classes	Architecture
Yale-B	5760	1200	10	1200-500-100-30
ORL	400	644	40	644-500-100-30
COIL-20	1440	1024	20	1024-500-250-100
COIL-100	7200	1024	100	1024-500-250-100
USPS	5000	256	10	256-400-200-100
DIGITS	5000	1024	10	1024-500-200-100

B. Evaluation Metric

As the label of each data point is available, we can compare clustering results with these labels to evaluate the performance. Here we use Normalized Mutual Information (NMI) and Accuracy (ACC) [5][22] to measure the effectiveness of clustering methods.

Normalized Mutual Information is a popular metric used for evaluating cluster tasks. It is defined as follows:

$$NMI(C, C') = \frac{I(C,C')}{max(H(C),H(C'))} \tag{20}$$

where C denotes clustering label and C' denotes ground truth label. $I(C, C')$ is mutual information which measures the

[1]http://markus-breitenbach.com/machine_learning_data.php
[2]http://www.cs.columbia.edu/CAVE/software/softlib/coil-20.php
[3]http://www.cs.columbia.edu/CAVE/software/softlib/coil-100.php
[4]http://archive.ics.uci.edu/ml

TABLE II. COMPARISON OF NMI AND ACCURACY OF CLUSTERING METHOD ON SIX DATASETS

Method	Yale-B		ORL		COIL20		COIL100		USPS		DIGITS	
	NMI(%)	*ACC(%)*	*NMI(%)*	*ACC(%)*	*NMI(%)*	*ACC(%)*	*NMI(%)*	*ACC(%)*	*NMI(%)*	*ACC(%)*	*NMI(%)*	*ACC(%)*
N-GNMF	**91.22±3.14**	87.63±5.73	**87.05±0.87**	**73.42±2.69**	88.87±1.49	**79.28±3.85**	**86.35±0.66**	**65.71±2.02**	**87.63±0.69**	**93.85±1.28**	**87.39±1.35**	**91.5±3.1**
N-NMF	89.79±1.23	**88.19±3.18**	85.07±1.28	68.43±2.84	78.13±1.2	67.75±2.48	77±0.52	53.12±1.39	71.39±0.04	78.31±0.05	67.93±0.34	76.56±0.85
GNMF	64.97±2.08	59.14±2.91	83.19±1.13	67.86±2.52	**89.43±0.88**	78.19±2.76	80.81±0.93	58.26±1.72	84.02±1.04	85.28±3.6	86.9±2.06	87.75±3.7
NMF	72.32±1.13	60.63±1.39	80.77±1.27	63.55±2.51	74.12±1.46	65.04±2.74	72.96±0.44	46.89±1.21	62.44±0.04	75.13±0.04	63.58±0.06	69.15±0.11
K-Means	85.72±1.74	79.48±1.91	81.81±1.21	64.48±2.54	74.18±1.33	62.57±2.93	73.52±0.58	46.93±1.39	65.6±1.47	71.89±4.08	73.33±0.49	77.84±0.69

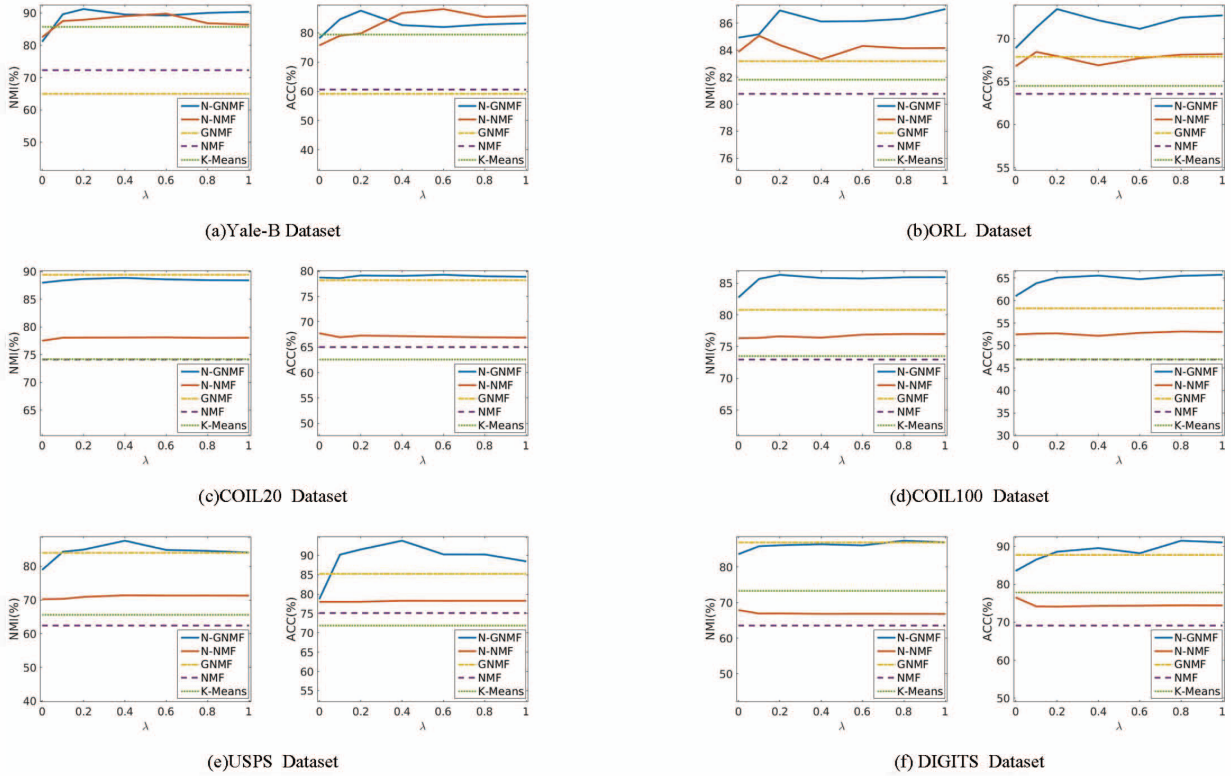

Fig. 3. NMI and Accuracy v.s. the parameter λ on six datasets

information gain to the true partition after knowing the clustering result, $H(C)$ and $H(C')$ are the entropies of C and C' respectively, and the $max(H(C), H(C'))$ is used to normalize the mutual information to be in the range of [0,1]. When the two sets of clusters are identical, NMI score is 1, and when the two sets are independent, NMI score is 0.

Accuracy is defined as:

$$ACC = \frac{\sum_{i=1}^{m} \delta(map(c_i)=y_i)}{m} \tag{21}$$

where y_i is the true group label of $\boldsymbol{v}^i$, c_i is the clustering label for $\boldsymbol{v}^i$, $\delta(x, y)$ is the delta function that equals 1 if $x = y$ and equals 0 otherwise, and $map(\cdot)$ transforms the clustering label c_i to its group label by the Hungarian algorithm [23].

C. Result Comparisons

To demonstrate how the clustering performance can be improved by our method, we compared our algorithm with k-means, classical NMF and GNMF. After our algorithms and classical NMF and GNMF algorithm obtain the factor $\boldsymbol{X}$, we employ k-means algorithm to perform clustering. Initial centroids impact on the clustering results when utilizing the k-means algorithm. We repeat the k-means algorithm 100 times with random initial centroids.

Our algorithms iterate the training process 300 loops and the comparison of results is listed in Table II. Our algorithms get the best clustering results, expect the NMI result on the COIL20 dataset. It is sufficient to extract nonlinear structure with graph regularization on COIL20 and DIGITS dataset, and the results of N-GNMF and GNMF are similar. Deep network can improve or reduce the clustering result slightly on the two datasets. However, graph regularization can't meet to all the datasets (e.g. Yale-B dataset) with different structure. At this time, we can find suitable nonlinear structure in the raw data using deep learning. Our experiments indicate our algorithm improves the clustering effect on almost all of the datasets.

Fig. 3 shows evaluation results in six different datasets (Yale-B, ORL, COIL20, COIL100, USPS, DIGITS) when setting the reconstruction error weight λ from among the values

[0, 0.1, 0.2, 0.4, 0.6, 0.8, 1]. Overall, we observe that the algotithms show good performance with the parameter λ in the range from 0.2 to 0.6.

VII. Conclusions

In this paper, the stacked auto-encoder is embedded into the framework of NMF to realize the task-specific nonlinear dimensional reduction. We designed the nonlinear NMF and GNMF optimization model and developed the optimization algorithm. The obtained representations are used for the task of clustering and the results show superior performance.

Acknowledgment

This work was completed while the first author was visiting Tsinghua University and TNLIST. This work was supported in part by the National Key Project for Basic Research of China under Grant 2013CB329403, in part by the National Natural Science Foundation of China under Grant 61327809, and in part by the National High-Tech Research and Development Plan under Grant 2015AA042306.

References

[1] S. Yazawa, M. Hamanaka, and T. Utsuro, "Novel approach to separation of musical signal sources by NMF," 12th International Conference on Signal Processing (ICSP), 2014, pp. 610-615.

[2] T. X. Luong, B. K. Kim, and S. Y. Lee, "Color image processing based on Nonnegative Matrix Factorization with Convolutional Neural Network," International Joint Conference on Neural Networks (IJCNN), 2014, pp. 2130-2135.

[3] Y. Yang, and B. Hu, "Pairwise Constraints-Guided Non-negative Matrix Factorization for Document Clustering," IEEE/WIC/ACM International Conference on Web Intelligence, 2007, pp. 250-256.

[4] D. D. Lee, and H. S. Seung, "Algorithms for Non-negative Matrix Factorization." Advances in Neural Information Processing Systems. 2001.

[5] D. Cai, X. He, X. Wu, and J. Han, "Non-negative Matrix Factorization on Manifold." IEEE International Conference on Data Mining (ICDM) , 2008, pp. 63-72.

[6] D. Zhang, and W. Liu, "An efficient nonnegative matrix factorization approach in flexible kernel space," in Proc. International Joint Conference on Artificial Intelligence, 2009, pp. 1345-1350.

[7] V. Duong, W. Hsieh, P. T. Bao, and J. Wang, "An overview of kernel based nonnegative matrix factorization," IEEE International Conference on Orange Technologies (ICOT) , 2004, pp. 227-231.

[8] S. Gao, Y. Zhang, K. Jia, J. Lu, and Y. Zhang, "Single Sample Face Recognition via Learning Deep Supervised Autoencoders," IEEE Transactions on Information Forensics and Security, vol.10, no.10, pp. 2108-2118, 2015.

[9] R. Salakhutdinov, and G. E. Hinton, " Learning a Nonlinear Embedding by Preserving Class Neighbourhood Structure," in Proc. International Conference on Artificial Intelligence and Statistics, 2007, pp. 412-419.

[10] P. Huang, Y. Huang, W. Wang, and L. Wang, "Deep Embedding Network for Clustering," International Conference on Pattern Recognition (ICPR) , 2014, pp. 1532-1537.

[11] H. Tseng, M. Hong, and Z. Luo, "Combining sparse NMF with deep neural network: A new classification-based approach for speech enhancement," IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), 2015, pp. 2145-2149.

[12] J. T. Geiger, F. Weninger, J. F. Gemmeke, M. Wollmer, B. Schuller, and G. Rigoll, "Memory-Enhanced Neural Networks and NMF for Robust ASR," IEEE/ACM Transactions on Audio, Speech, and Language Processing, vol.22, no.6, pp. 1037-1046, 2014.

[13] D. S. Williamson, Y. Wang, and D. Wang, "Deep neural networks for estimating speech model activations," IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), 2015, pp. 5113-5117.

[14] T. G. Kang, K. Kwon, J. W. Shin, and N. S. Kim, "NMF-based Target Source Separation Using Deep Neural Network," IEEE Signal Processing Letters, vol.22, no.2, pp. 229-233, 2015.

[15] E. M. Grais, M. U. Sen, and H. Erdogan, "Deep neural networks for single channel source separation," IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), 2014, pp. 3734-3738.

[16] J. L. Roux, J. R. Hershey, F. Weninger, "Deep NMF for speech separation Acoustics," IEEE International Conference on Speech and Signal Processing (ICASSP), 2015, pp. 66-70.

[17] I. Redko, Y. Bennani, "Sparsity analysis of learned factors in Multilayer NMF," International Joint Conference on Neural Networks (IJCNN), 2015, pp. 1-7.

[18] G. E. Hinton, and R. R. Salakhutdinov, "Reducing the Dimensionality of Data with Neural Networks," Science, vol. 313. no. 5786, pp. 504-507, 2006.

[19] A. S. Georghiades, P. N. Belhumeur, and D. J. Kriegman, "From few to many: Illumination cone models for face recognition under variable lighting and pose," IEEE Transactions on Pattern Analysis and Machine Intelligence, vol.23, no.6, pp. 643-660, 2001.

[20] F. S. Samaria, and A. C. Harter, "Parameterisation of a stochastic model for human face identification," in Proceedings of the Second IEEE Workshop on Applications of Computer Vision, 1994, pp. 138-142.

[21] J. J. Hull, "A database for handwritten text recognition research," IEEE Transactions on Pattern Analysis and Machine Intelligence, vol.16, no.5, pp. 550-554, 1994.

[22] W. Xu, X. Liu, and Y. Gong, "Document clustering based on non-negative matrix factorization," In Proc. Int. Conf. on Research and Development in Information Retrieval (SIGIR) , 2003, pp. 267–273.

[23] C. H. Papadimitriou and K. Steiglitz, Combinatorial optimization: algorithms and complexity, Courier Dover Publications, 1998.